

# NAVAL POSTGRADUATE SCHOOL

## Monterey, California



DTIC QUALITY INSPECTED 2

## THESIS

**REENGINEERING THE UNITED STATES MARINE  
CORPS' RECRUIT DISTRIBUTION MODEL (RDM)**

by

Kevin J. Snoap

September 1998

Thesis Advisor:  
Associate Advisor:

Hemant K. Bhargava  
Suresh Sridhar

**Approved for public release; distribution is unlimited.**

19981009 117

# REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.

<b>1. AGENCY USE ONLY (Leave blank)</b>		<b>2. REPORT DATE</b> September 1998	<b>3. REPORT TYPE AND DATES COVERED</b> Master's Thesis	
<b>4. TITLE AND SUBTITLE : REENGINEERING THE UNITED STATES MARINE CORPS' RECRUIT DISTRIBUTION MODEL (RDM)</b>			<b>5. FUNDING NUMBERS</b>	
<b>6. AUTHOR(S)</b> Kevin J. Snoap				
<b>7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)</b> Naval Postgraduate School Monterey, CA 93943-5000			<b>8. PERFORMING ORGANIZATION REPORT NUMBER</b>	
<b>9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)</b>			<b>10. SPONSORING / MONITORING AGENCY REPORT NUMBER</b>	
<b>11. SUPPLEMENTARY NOTES</b> The views expressed in this thesis are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government.				
<b>12a. DISTRIBUTION / AVAILABILITY STATEMENT</b> Approved for public release; distribution is unlimited.			<b>12b. DISTRIBUTION CODE</b>	
<b>13. ABSTRACT (maximum 200 words)</b> The United States Marine Corps accomplishes its mission "to put the right Marine in the right place at the right time with the right skills and quality of life" in a variety of ways. One of the information systems assisting the Marine Enlisted Assignments branch is the Recruit Distribution Model (RDM). This thesis proposes changes to the RDM user interface, data management, assignment model, and analysis capability. With the use of business process reengineering, process modeling, mathematical modeling, and database design a fully functional prototype has been developed to address each identified change proposal. This reengineered system includes numerous innovations such as an intuitive navigational scheme using switchboards, and the elimination of manual data entry for data already available in the system. It also provides a number of significant contributions beneficial to the USMC. For instance, the reengineered system allows the user to objectively analyze different results by comparing four different objective measures, and its mathematical model uses commercial-off-the-shelf products eliminating a proprietary solver. All these changes will empower managers to effectively and efficiently manage the assignment of recruits in order to meet the challenges of the 21st century.				
<b>14. SUBJECT TERMS</b> USMC, Databases, Manpower Assignment, Models, Decision Support Systems, Graphical User Interface			<b>15. NUMBER OF PAGES</b> 100	
			<b>16. PRICE CODE</b>	
<b>17. SECURITY CLASSIFICATION OF REPORT</b> Unclassified	<b>18. SECURITY CLASSIFICATION OF THIS PAGE</b> Unclassified	<b>19. SECURITY CLASSIFICATION OF ABSTRACT</b> Unclassified	<b>20. LIMITATION OF ABSTRACT</b> UL	

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)  
Prescribed by ANSI Std. Z39-18



Approved for public release; distribution is unlimited.

## REENGINEERING THE UNITED STATES MARINE CORPS' RECRUIT DISTRIBUTION MODEL (RDM)

Kevin J. Snoap  
Lieutenant, United States Navy  
B.S., University of Texas at Austin, 1991

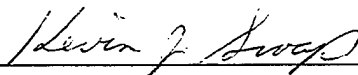
Submitted in partial fulfillment of the  
requirements for the degree of

MASTER OF SCIENCE IN INFORMATION TECHNOLOGY  
MANAGEMENT

from the

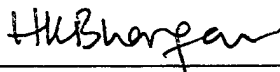
NAVAL POSTGRADUATE SCHOOL  
September 1998

Author:

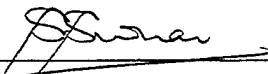


Kevin J. Snoap

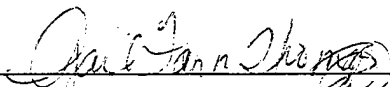
Approved by:



Hemant K. Bhargava, Advisor



Suresh Sridhar, Associate Advisor



Rueben T. Harris, Chairman, Department of Systems  
Management



# ABSTRACT

The United States Marine Corps accomplishes its mission "to put the right Marine in the right place at the right time with the right skills and quality of life" in a variety of ways. One of the information systems assisting the Marine Enlisted Assignments branch is the Recruit Distribution Model (RDM). This thesis proposes changes to the RDM user interface, data management, assignment model, and analysis capability. With the use of business process reengineering, process modeling, mathematical modeling, and database design a fully functional prototype has been developed to address each identified change proposal. This reengineered system includes numerous innovations such as an intuitive navigational scheme using switchboards, and the elimination of manual data entry for data already available in the system. It also provides a number of significant contributions beneficial to the USMC. For instance, the reengineered system allows the user to objectively analyze different results by comparing four different objective measures, and its mathematical model uses commercial-off-the-shelf products eliminating a proprietary solver. All these changes will empower managers to effectively and efficiently manage the assignment of recruits in order to meet the challenges of the 21st century.



# TABLE OF CONTENTS

<b>I.</b>	<b>INTRODUCTION . . . . .</b>	<b>1</b>
A.	THESIS PURPOSE . . . . .	1
B.	GENERAL PROBLEM DESCRIPTION . . . . .	1
C.	SIGNIFICANT CONTRIBUTIONS . . . . .	5
<b>II.</b>	<b>REENGINEERING MOTIVATIONS AND PROCESS VIEW</b>	<b>7</b>
A.	LIMITATIONS OF OLD SYSTEM . . . . .	7
1.	Navigation . . . . .	7
2.	Data Management . . . . .	8
3.	Assignment Procedure . . . . .	8
4.	Transaction Processing Approach . . . . .	9
B.	A PROCESS VIEW . . . . .	9
1.	Recruit Distribution Model Operating Environment . . . . .	10
2.	New System Level 0 Diagram . . . . .	11
<b>III.</b>	<b>MODELING THE RECRUIT DISTRIBUTION PROBLEM .</b>	<b>15</b>
A.	OPTIMIZATION OBJECTIVES: FIT AND FILL . . . . .	15
B.	MEASURING FITNESS . . . . .	16
C.	MEASURING FILL . . . . .	18
D.	ASSIGNMENT MODEL . . . . .	19
1.	Assumptions . . . . .	19
2.	Notation . . . . .	20
3.	Objective . . . . .	20
4.	Constraints . . . . .	20
<b>IV.</b>	<b>A DECISION SUPPORT SYSTEM FOR RECRUIT DISTRI- BUTION . . . . .</b>	<b>23</b>
A.	ARCHITECTURE AND IMPLEMENTATION . . . . .	23
1.	Architecture . . . . .	23



2.	Implementation . . . . .	26
B.	USING THE RECRUIT DISTRIBUTION DECISION SUPPORT SYSTEM . . . . .	28
1.	Setting up a Run . . . . .	28
2.	Model Execution . . . . .	31
3.	Customizing a Run and Results . . . . .	33
C.	INNOVATIONS IN USER INTERACTION . . . . .	34
D.	INNOVATIONS IN ANALYZING MODEL RESULTS . . . . .	35
E.	OBJECTIVE COMPARISON OF OLD AND NEW SYSTEM SOLUTIONS . . . . .	39
F.	SUMMARY . . . . .	41
V.	CONCLUSION . . . . .	43
A.	SIGNIFICANT CONTRIBUTIONS . . . . .	43
B.	LESSONS LEARNED . . . . .	43
C.	PROTOTYPE IMPROVEMENTS . . . . .	44
1.	Speed Improvements . . . . .	44
2.	Multiple Solution Storage . . . . .	45
3.	Administration Switchboard . . . . .	46
	APPENDIX A. ACRONYMS . . . . .	47
	APPENDIX B. AS-IS RD BUSINESS PROCESS (IDEF0) MODEL	49
	APPENDIX C. TO-BE RD BUSINESS PROCESS (IDEF0) MODEL	61
	APPENDIX D. RDDSS VBA CODE . . . . .	67
	LIST OF REFERENCES . . . . .	87
	INITIAL DISTRIBUTION LIST . . . . .	89

# LIST OF FIGURES

1.	Assigning Marines to Schools . . . . .	2
2.	Assigning Marines to School Classes . . . . .	4
3.	Current Recruit Distribution Operating Environment . . . . .	10
4.	RDdss Process, Level 0 Diagram . . . . .	12
5.	Exponential Function for Calculating $Score_{level(p)}$ . . . . .	17
6.	Two Candidate Penalty Functions . . . . .	19
7.	RDdss Architecture . . . . .	23
8.	RDdss Main Switchboard . . . . .	24
9.	RDdss Relational Schema . . . . .	25
10.	RDdss Demonstration Solver . . . . .	26
11.	RDdss Decomposition Diagram . . . . .	27
12.	Actual RDdss Build Sequence . . . . .	28
13.	Setting up a Model Run . . . . .	29
14.	Model Execution . . . . .	32
15.	Changing a Graduation Date . . . . .	35
16.	Two alternative results: Fit . . . . .	37
17.	Two alternative results: Fill . . . . .	37



# **I. INTRODUCTION**

## **A. THESIS PURPOSE**

The main purpose of this thesis was to reengineer a United States Marine Corps' Manpower Assignment model concerned with the distribution of recruits to schools. This model is called the Recruit Distribution Model (RDM) [Ref. 1]. Throughout this thesis, the RDM is addressed as either RDM or the old system.

Additionally, an important purpose of this thesis was to build a functional prototype of the reengineered RDM. The new system is called the Recruit Distribution Decision Support System (RDdss). It demonstrates the functionality of the reengineered RDM. Throughout this thesis, the RDdss is addressed as either RDdss or the new system.

The majority of this thesis is devoted to a discussion of the RDdss. As necessary, the RDM is discussed. The following section is a general discussion designed to set the stage for understanding the rest of the thesis. It provides a problem description of recruit distribution in the United States Marine Corps (USMC). The last section of this chapter discusses the significant contributions of this thesis.

## **B. GENERAL PROBLEM DESCRIPTION**

Recruit distribution in the USMC is the process that assigns recruits to an entry level school (ELS) leading to a military occupational specialty (See Figure 1). These assignments are made about 48 times a year, during the last week of Marine Corps Recruit Depot (MCRD) training. In this ending period of the MCRD, the recruits are facing the "Crucible," which is the final wicket a recruit endures before officially becoming a Marine. Consequently, the use of the titles recruit and Marine are used interchangeably in this paper.

For at least two reasons, this assignment process is a critical manpower function. First, a Marine's military occupational specialty (MOS) ultimately determines

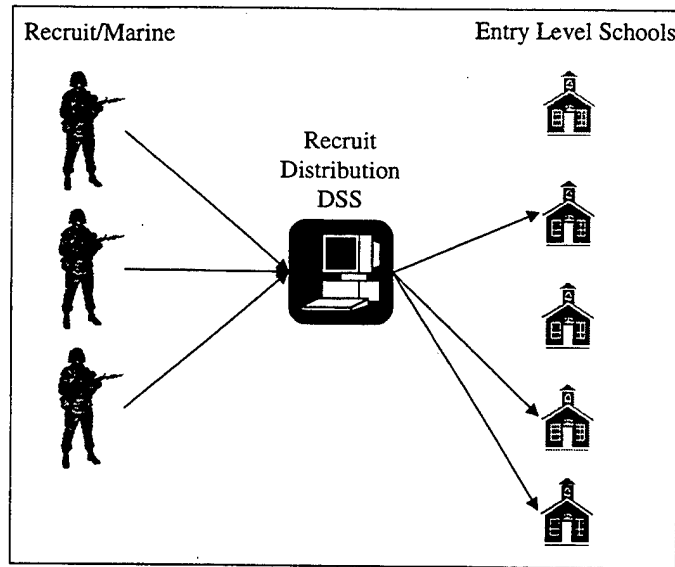


Figure 1. Assigning Marines to Schools

the member's career. Therefore, it is in the best interest of both the Marine and the USMC that a school assignment matching the Marine's desire is made. Second, the success obtained by a service member during his or her time in the USMC is partially based upon successful completion of their ELS, where a pattern of success is established. Therefore, a school assignment maximizing the chances of the Marine completing their training is important. So, fulfilling the Marine's desire and matching him or her to an MOS (i.e. ELS) that fits their personal characteristics is critical to the overall health of the USMC, making the assignment process a critical manpower function [Ref. 2].

Fulfilling the desire of the Marine is accomplished through the use of a contract guarantee. This is called a program enlisted for (PEF), and is specified during the recruiting process. For instance, a PEF = 19 is the "Tank and Assault Amphibian Option." There are currently two ELS's associated with this, "M1A1 Tank Crewman" and "Assault Amphibian Crewman." So, a Marine who chose the PEF = 19 is a possible candidate for these two schools and no others.

Once the Marine's school options are known, a Marine-to-school fit is deter-

mined for each of these schools (See Chapter III, Section B for details). This fitness determination is partly made by looking at each ELS's minimum eligibility requirements, which are called mandatory properties. This term "property" is used as defined by Webster's Ninth New Collegiate Dictionary: "a quality or trait belonging and especially peculiar to an individual or thing." Examples are Age  $> 18$ , Clerical  $\geq 80$ , and Electrical  $\geq 95$ . The meaning of the first example is obvious. The other two are based on test scores from the Armed Services Vocational Aptitude Battery (ASVAB) test.

In addition to the mandatory properties, most schools also specify desirable properties. A desired property is the same as a mandatory property, except they are not prerequisites for attending the school. For example, the Traffic Management Coordination school desires Marines with a Clerical score of at least 100. So, a desired property of Clerical  $\geq 100$  is specified for this ELS.

By using the information obtained from the PEF, mandatory and desired properties, a fitness matrix is generated. This shows the fit of every Marine to every school he or she is eligible for. Since there are about 100 schools and on average 700 Marines considered for every run, this matrix has the potential of 70,000 matches.

However, the matrix size is actually bigger. This is because each of the schools is broken down by classes. Some ELS's have a class starting each week, others every month, and others every quarter. Over a given year, these classes total about 1,800. Following the practice of the USMC, only the classes over the next 3-4 months are considered during the assignment process (see Figure 2). Therefore, the fitness matrix is increased to a potential size of 350,000.

The fact that the school classes start at different times throughout the year and that the assignment process is only conducted 48 times a year (normally this occurs on Friday) causes the "3-month look ahead" in the model and has implications important to the USMC. The concern is school seats which never get filled. Each seat is prepaid, guaranteeing its availability to the USMC. This means every vacant

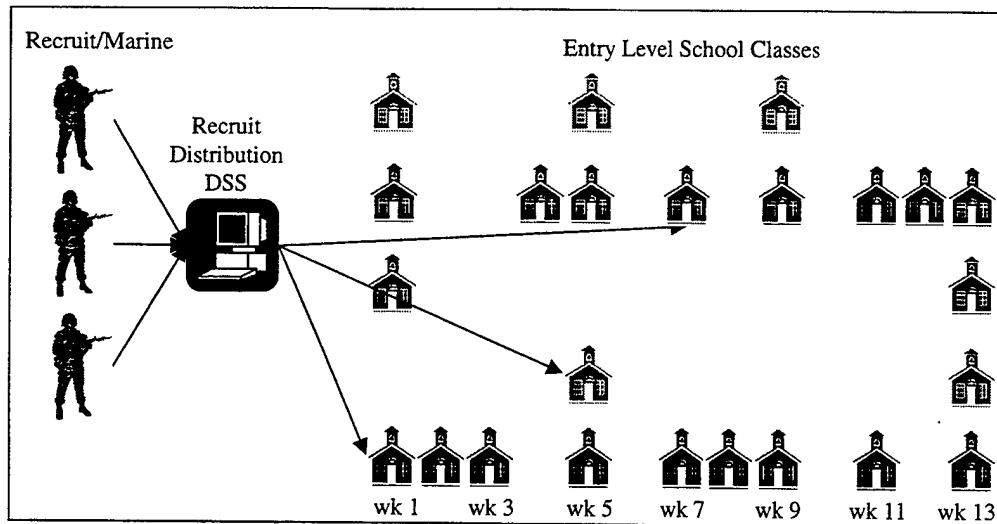


Figure 2. Assigning Marines to School Classes

spot is potentially a lost resource to the USMC. Therefore, in addition to the Marine-to-school fitness concern, there is also the concern of filling all available school seats before their report date is passed.

Another concern is the problem of unassigned Marines. There are numerous reasons why they may not get assigned. Maybe the only school class the Marine was eligible for is already full. Or, possibly he or she is not qualified for any of the school's promised by their PEF. A third possibility is errors in the data provided to the assignment process. Regardless of the cause, it requires identification and corrective action.

Finally, a discussion of the internal and external stakeholders associated with the USMC recruit distribution is given. Internally, there are the USMC and the recruits. Since their concerns were discussed above, they are not given any further consideration. Externally, there is the contractor Decision Support Associates, Inc. (DSAI) who has maintained and upgraded the RDM for over 30 years. They also maintain about eight other major systems for the USMC. For their services, the USMC pays a significant amount of money each year. Additionally, DSAI has proprietary software in some of these systems, which has locked-in the USMC to this

company. Besides the contractor, there is one last significant stakeholder, the American taxpayer. In a time of increasing fiscal constraints and shrinking budgets, it is imperative that wise decisions are made in regards to assigning Marines to ELS's. The American taxpayers expect nothing less.

## **C. SIGNIFICANT CONTRIBUTIONS**

Four significant contributions beneficial to the USMC have been made in this thesis. A detailed discussion of these is given throughout the next three chapters. Here, we list and summarize the contributions.

- Analysis and articulation of the recruit distribution process - by interviewing USMC and DSAI personnel, reviewing available documentation, and operating the RDM we were able to articulate an understanding of the recruit distribution process using IDEF process modeling. Additional data modeling was articulated in a third normal form relational schema [Ref. 3].
- Development of a mathematical model - by analyzing the assignment process, criteria, and constraints, USMC policies and objectives, a mathematical model for the new system was developed.
- Fully functional prototype - an intuitive and easy to understand new system that seamlessly interfaces with the old system was built. It provides an objective means of comparing assignment results of both systems, and was developed using commercial-off-the-shelf (COTS) software applications.
- Elimination of the proprietary solver and associated contractor lock-in - this was accomplished by replacing the proprietary solver with two COTS applications.





## **II. REENGINEERING MOTIVATIONS AND PROCESS VIEW**

The RDM has a number of limitations which motivated the USMC to reengineer it. We start this chapter by listing and describing each identified limitation. Then, to develop a better understanding of recruit distribution in the USMC, two different process views are examined. The first is concerned with the RDM operating environment, and the second is concerned with the first level of the RDdss IDEF0 model.

### **A. LIMITATIONS OF OLD SYSTEM**

Throughout this reengineering effort, a number of limitations of the old system have become apparent. Many of these limits were recognized earlier, and were part of the USMC's motivation to reengineer the RDM. The following is a list of the identified limitations of the old system.

- Navigation
- Data management
- Assignment procedure
- Transaction processing approach

A description of each limitation is now given.

#### **1. Navigation**

Navigating through the RDM is not intuitive. It is neither obvious where one should start or where one should go. Navigation is accomplished by initially selecting an option from one of the main display's drop down menus. This normally results in a window appearing on the computer's desktop. Then, by pointing and clicking on the displayed window's buttons, further navigation is accomplished. After working with the RDM for a number of hours, we were able to navigate through the application

to find and display specific windows. However, after a week or two of not using the system, we had difficulty finding our way around again. In the RDdss, we have created an intuitive navigational scheme using switchboards.

## **2. Data Management**

Data management in the RDM is poor, leading to the introduction of numerous errors. The biggest problem in this regard deals with data entry. The RDM violates the basic rule of never requiring the user to enter data already in the system [Ref. 4]. For example, creating a new school in the RDM requires the user to type data into seven different data fields. Only the course identification number has been automated by a drop down list. In the RDdss, when creating a new school we have reduced the manual data entry to one field.

## **3. Assignment Procedure**

There are at least five limitations associated with the old system's assignment procedure. It

- is encoded into proprietary software,
- examines schools sequentially, rather than globally,
- makes assignments based on school priority rather than weights,
- attempts to maximize fill rather than fit-and-fill, and
- is relatively inflexible.

The solver performing these assignments was designed back in the 1950's, where the major concern was speed and using the minimal amount of memory. It is written in Fortran, and is proprietary code owned by DSAI. It does not search repetitively for an optimal solution by trying to maximize or minimize an objective function. Instead, it maximizes the fill of prioritized school seats. In the RDdss, we use a well know algorithm called CPLEX [Ref. 5]. After conducting around 600 iterations of comparing 344 Marines to 576 school classes, it produces an *optimal* solution. Its

objective function was written to make the assignment procedure flexible. It allows the RDdss manager to “game” the system by making fit-and-fill trade-offs, until a “good” solution is found.

#### **4. Transaction Processing Approach**

Finally, the old system follows a transaction processing approach, vice a decision support approach [Ref. 6]. This is partially due to the inflexibility of the solver. However, another contributor is the extreme difficulty in comparing one run to another. Other than providing a means for manually computing the numerical difference in Marines assigned, the RDM provides little insight or information views for comparing runs. This is because there is no way to objectively compare one run to another. As it is now, if the RDM manager launches the assignment model and everybody is assigned, the result becomes the approved assignments. In the RDdss, we have created an entire process devoted to providing insightful analysis of a given run (See Chapter IV, Section D for details). Its purpose is to support the making of wise assignment decisions.

### **B. A PROCESS VIEW**

To develop a better understanding of recruit distribution in the USMC, two different process views are now examined. The first is concerned with the current operating environment of the old system. The goal is to develop a big picture view of this process. For those interested in further study of the old system, Appendix A contains the RDM business process IDEF0 model.

The second process view examined concerns the first level of the RDdss business process IDEF0 model. The goal is to develop a better understanding of the new system, without going into great detail. This lays a good foundation for the RDdss discussions in the remainder of this thesis. Appendix B contains the entire IDEF0 model of the new system.

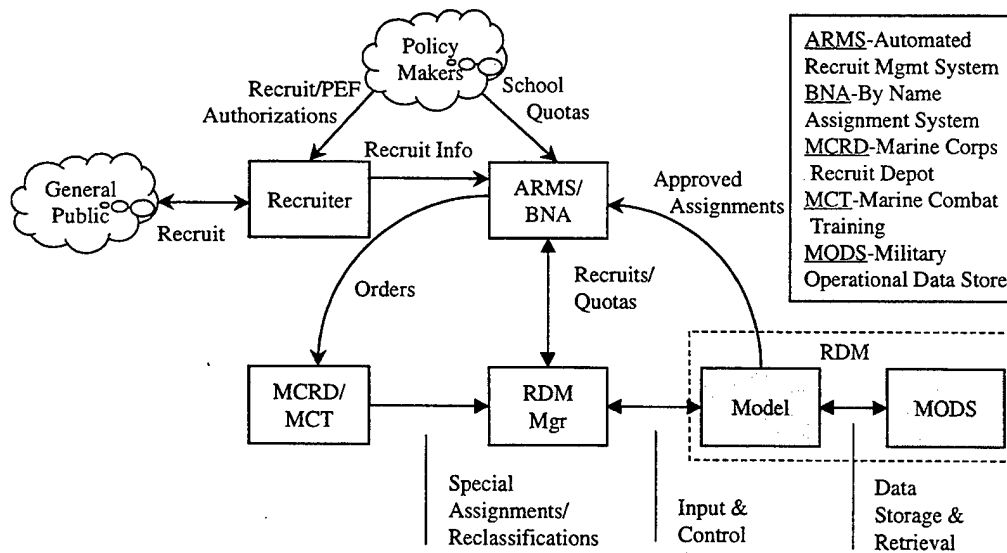


Figure 3. Current Recruit Distribution Operating Environment

## 1. Recruit Distribution Model Operating Environment

Figure 3 graphically depicts the operating environment of the USMC recruit distribution. At the top are the policy makers, who forecast and decide how many recruits and corresponding school seats are needed over the next few years. In addition to this, they determine how many PEF's or school guarantees to make available for a given year. Currently, about 65-70% of the recruits enter the USMC with a guarantee. The recruit and PEF authorizations for the following year are given to the USMC recruiters, and the school seats or quotas determined for the following years is input into the By Name Assignment (BNA) system.

The recruiters use the recruit and PEF authorizations in recruiting from the general public. Once a potential candidate is found they take the ASVAB, if they have not already done so. Based on the results, the recruiter can offer different guarantees. Once the candidate signs a contract with the USMC, their test scores, PEF, and other personal information such as age and height are entered into the Automated Recruit Management System (ARMS).

Both the ARMS and BNA systems utilize large main frame databases. They

serve as central repositories for maintaining data on many aspects dealing with the USMC. Keeping this information stored in one location ensures everyone is using the same data, providing consistency for all users. One user is the RDM manager, who retrieves recruit and school information from these systems for use in the RDM.

Additionally, the RDM receives data from two other sources. The MCRD instructors provide special assignment inputs. These are personnel identified as having the talents or abilities well suited for a particular school. The other data comes from the MCT. They provide the RDM manager with reclassification information. For instance, a Marine is reclassified if he or she is injured during MCT and is unable to make the start date of their assigned ELS.

All this recruit and school data is input into the RDM, where it is stored in the Military Operational Data Store (MODS). The model is then run. Once a satisfactory set of assignments is obtained, the RDM manager uploads the approved assignments to the ARMS and BNA systems. From this assignment information, the MCRD generates orders for the Marines graduating from the MCRD.

## **2. New System Level 0 Diagram**

Focus is now shifted to the examination of the first process level of the RDdss. Looking any deeper will provide more detail than is necessary at this point. As mentioned earlier, the intent is to lay the foundation for the RDdss discussions in the remainder of this thesis.

Figure 4 shows the level 0, or context diagram of the RDdss. The program used to generate this illustration was BPWin, which is based on IDEF0 modeling. Consequently, all arrows entering from the left are considered "inputs," arrows entering from the top are "controls," arrows exiting to the right are "outputs," and arrows entering from the bottom are "mechanisms." A convenient acronym for this is ICOM [Ref. 7].

- I = Input: something consumed in the process
- C = Control: a constraint on the operation of the process

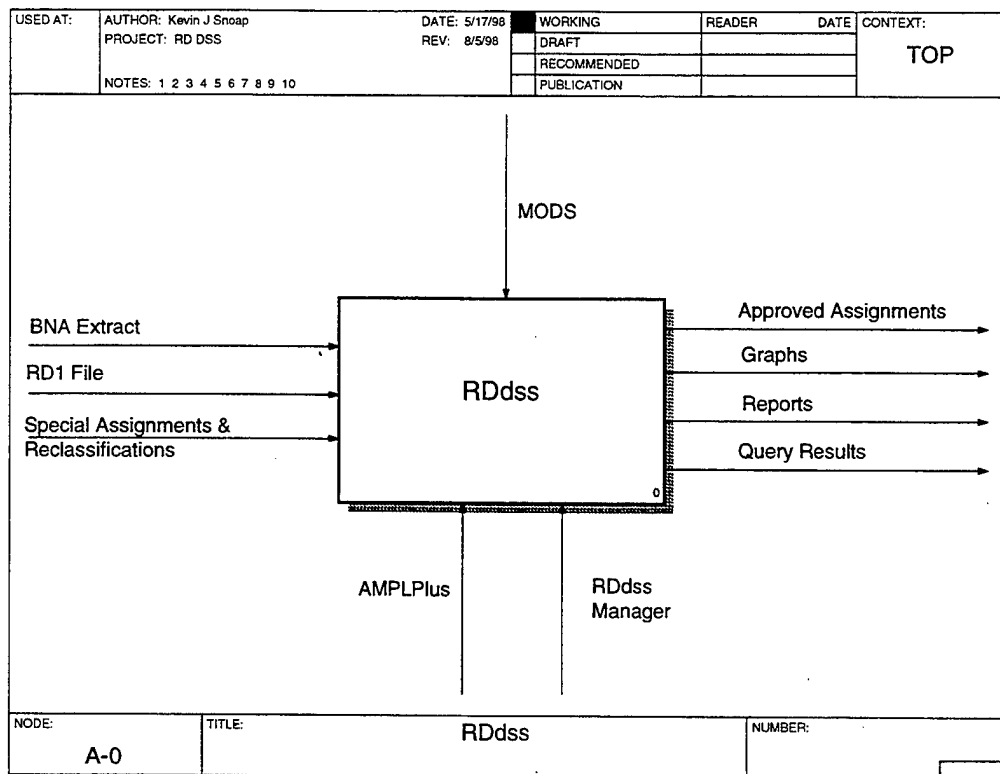


Figure 4. RDdss Process, Level 0 Diagram

- O = Output: something resulting from the process
- M = Mechanism: something used to perform the process, but is not itself consumed

There are three inputs to the RDdss, the BNA extract, the RD1 file, and the special assignments/reclassifications. The first of these contains a set of school class quotas covering a 120 day period. This comes to the RDdss manager as a fixed-width delimited text file. The RD1 file contains the data for the recruits soon to graduate from the MCRD. It also is a fixed-width delimited text file. The special assignments and reclassifications are the recruits identified with special abilities or talents and the Marines requiring reassignment to a different school class, respectively. This last data comes to the RDdss manager as a memo and not a text file.

The only control is the database (or Military Operational Data Store) used to store the data for the RDdss. Among other things, the MODS contains the informa-

tion specifying the eligibility requirements for each of the schools. This is why it is considered a control. It contains the data controlling or constraining the assignment of Marines to schools.

Besides the approved assignments that were mentioned earlier, the output consists of graphs, reports, and query results. Each graph was designed to provide insight into the current assignment result, plus provide a means for comparing different runs. The reports provide a print-out of information the RDdss manager might find useful. For instance, an approved assignment report and an unassigned Marine report are both available. The query results provide different views of the data in the MODS. For instance, one of the queries provides a listing of the fitness scores of all schools a Marine is eligible for.

Finally, the two mechanisms of the RDdss are the RDdss manager and the commercial-off-the-shelf (COTS) application AMPLPlus. Both of these entities are necessary to make the system work, and are not consumed by the process itself. This is why they are considered mechanisms. The RDdss manager is the one who points and clicks on the buttons of the RDdss, making it operate. AMPLPlus is the application containing the mathematical model used in making optimal Marine-to-school assignments. It interfaces with the CPLEX algorithm that actually performs the optimization.





### III. MODELING THE RECRUIT DISTRIBUTION PROBLEM

We model the recruit distribution problem as an assignment problem. Various submodels are developed to compute input parameters and perform other preprocessing steps for the assignment model, which optimally assigns recruits to a school class. Following is a discussion of important points concerning the recruit distribution problem.

1. Schools and Classes: there are about 130 schools per fiscal year, which are broken down by classes (about 1800 classes per fiscal year). Classes of the same school, commencing on different dates, are identical. However, a class commencing soon after a recruit is available for training has greater utility than another class starting at a later date. On average, there are 600 classes in a run.
2. Program Enlisted For (PEF)
  - (a) 65-70% of enlistees enter the USMC under a guarantee (or PEF). The remaining recruits enter under an open contract (PEF=00).
  - (b) A recruit can be assigned only to schools associated with his or her PEF. Some PEF's are associated with one school, while others are associated with many schools.

#### A. OPTIMIZATION OBJECTIVES: FIT AND FILL

Our approach in the assignment model is to optimize by looking at both fit and fill. The importance of both these objectives was discussed in Chapter II, Section

B. The idea is to allow the RDdss manager to make a tradeoff between fit and fill.

The high-level objective function is:

- Maximize

$$K_1 \cdot \text{Fitness} - K_2 \cdot \text{Penalty}$$

Where  $K_1$  is the fitness coefficient and  $K_2$  is the fill coefficient.

By assigning different values to these coefficients, the RDdss manager is able to make trade-offs between fit and fill. This capability makes the assignment model flexible. It provides the RDdss manager a means for “gaming” the system.

## B. MEASURING FITNESS

The first half of the high-level objective function deals with maximizing the fitness of Marines to schools. Therefore, we cover this topic now. Following is a discussion of important points concerning the measuring of fitness.

1. Marines are only eligible for assignment to schools corresponding to their PEF guarantees.
2. Associated with each school are specific mandatory properties, or minimum eligibility requirements (e.g. AGE > 18).
3. Eligibility of a Marine to a school is pre-determined by comparing a Marine’s attributes (e.g. Age = 20) to the school’s mandatory properties. A Marine is eligible only if he or she meets all the minimum school requirements.
4. Many schools also have desirable properties associated with them. This is a property which is desired in a Marine (e.g. Height > 65 inches), but are not prerequisites for attending the school. Since some desirable properties are more desirable than others, a means of distinguishing the properties is necessary. We have followed the Marine Corps practice of using 6 levels of desirability. Level 1 properties are the most desired, followed by level 2, etc.
5. Let  $\mathcal{P}(s)$  denote the set of desirable properties for school  $s$ , and let  $level(p)$  denote the desirability level of property  $p$ . Let  $possesses_{(m,p)} = 1$  if Marine  $m$  possesses property  $p$  (0 otherwise). Further, let  $Score_{level(p)} (> 0)$  be the score assigned for possessing a  $level(p)$  property, where  $Score_{level(p)}$  is calculated using an exponential function inversely proportional to  $level(p)$ . The scores for levels 1 through 6 are shown in Figure 5.
6. The total fitness score ( $fit_{m,s}$ ) of Marine  $m$  to school  $s$  is composed of two parts: the score ( $ManScore_{m,s}$ ) for possessing mandatory properties, and the score ( $DesScore_{m,s}$ ) for possessing some or all of the desirable properties.

Our procedure for computing fitness scores is designed in a manner that, for each school, the average fitness score—with the average computed over all qualified Marines, i.e. Marines who have met the mandatory properties—is constant (i.e. 100). The reason we did this was to ensure each school

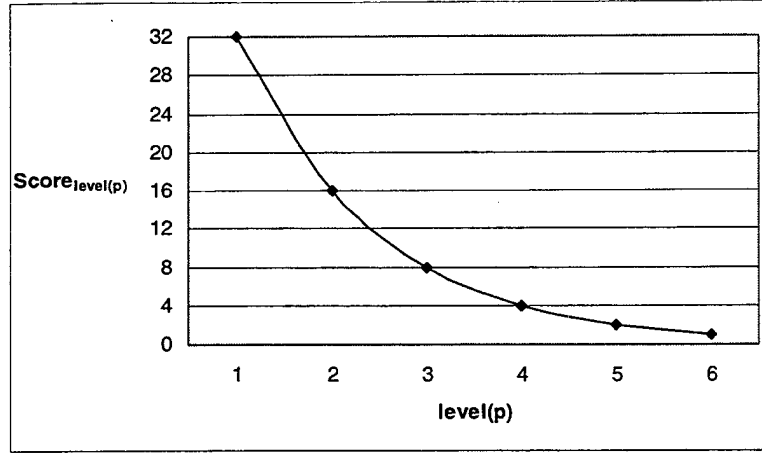


Figure 5. Exponential Function for Calculating  $Score_{level(p)}$

received equal treatment in the model, regardless of the number of desirable properties specified for each school. Otherwise, all other things being equal, the school with the greatest number of desirable properties would receive the most assignments.

7. For each school  $s$ , percentage weights  $ManWt_s$  and  $DesWt_s$  are assigned, respectively, for the mandatory and desirable properties. For example, for school  $s_1$ , a weight of 70% may be assigned to the mandatory properties and 30% for the desirable properties.
8. A Marine possessing all mandatory properties for school  $s$  is given an initial score  $ManScore_{m,s} = ManWt_s \cdot 100$ . Marines who do not possess all mandatory properties for a school are given an overall fitness score of 0 for that school.
9. Of all the Marines possessing the mandatory properties, those possessing desirable properties are awarded additional points, which are weighted by the level of the property. The Marine's score for desirable properties is computed as follows:
  - The absolute score  $Abs_{m,s}$ , for Marine  $m$  and school  $s$ , based on desirable properties is

$$Abs_{m,s} = \sum_{p \in \mathcal{P}(s)} Score_{level(p)} \cdot possesses(m, p)$$

The average of these absolute scores is computed over all qualified Marines. Let  $AveAbs_s$  denote the average absolute score for school  $s$ .

- Each Marine's fitness score for desirable properties is then computed as a fraction of this average, and normalized by multiplying with the percentage weight for desirable properties. That is,  $DesScore_{m,s}$  is a weighted relative score:

$$DesScore_{m,s} = \left( \frac{Abs_{m,s}}{AveAbs_s} \right) \cdot DesWt_s \cdot 100$$

The overall average of all these scores is, due to the above construction,  $DesWt_s \cdot 100$ .

10. The final fitness score,  $fit_{m,s}$  is simply the sum  $ManScore_{m,s} + DesScore_{m,s}$ . It may be seen that, for each school, this number averages (over all the qualified Marines for that school) to 100.
11. As mentioned earlier, schools are broken down by classes that are identical, except for their start dates. Therefore, let  $fit_{m,s} = fit_{m,c}$  for each class  $c$  of school  $s$ .

## C. MEASURING FILL

The second half of the objective function deals with maximizing the fill of school seats. We now turn our attention to this topic.

As we discussed in Chapter II, school seats are paid for in advance. This is to guarantee the seat is available to the USMC. Consequently, every vacant spot is potentially a lost resource. Since the model is run about once every week, the biggest concern is the unfilled seats having report dates within the next seven days. Conversely, a seat having a report date in three months is not so critical.

To capture the essence of filling school seats with early report dates, we use a penalty function. The idea is simple. School seats having an early report date get a high penalty, and those having a late report date get a low penalty. A number of functions would have worked for this. We have graphed two candidates in Figure 6. The first is a linear function and the second a large-step function. Each of these was tried in the prototype. In our opinion the latter one is the best choice. It gives a lot of emphasis to the first week, plus it treats all schools having report dates within the same week equally. It is the function currently implemented in the rddss.

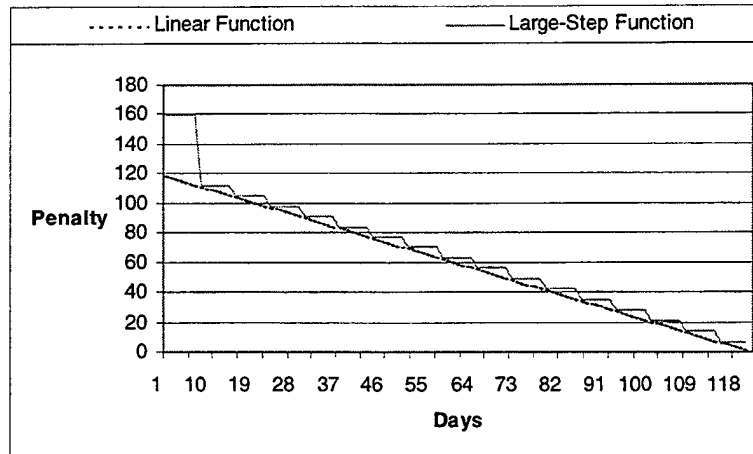


Figure 6. Two Candidate Penalty Functions

## D. ASSIGNMENT MODEL

We are now ready to formally describe the assignment model. We will first specify the model's assumptions. Then, we will list the model's notation, followed by its objective function and constraints.

### 1. Assumptions

1. Each school is unique and identifies a single course of instruction leading to an MOS.
2. In any given run, a school may offer several classes that start on different dates. It is more desirable to fill seats in classes starting earlier. Such classes have a higher penalty per unit shortfall than similar classes starting at a later date.
3. All Marines available for assignment are graduating from the MCRD on the same date.
4. A Marine's eligibility for a school, as well as the fit for each school, is determined by the model preprocessor, taking into account the PEF guarantees, and the mandatory and desirable properties.
5. A Marine is assigned to a school corresponding to their PEF code, or not at all.
6. It is better to leave a Marine unassigned than to overfill a school.

7. The demand for the number of Marines to be trained in each school class is determined by looking at the BNA extract. This demand statement is already constrained by the capacity constraint at each school.

## 2. Notation

- Sets

$\mathcal{M}$  : Marines

$\mathcal{C}$  : Classes

- Exogenous Variables (Parameters in AMPL)

$fit_{m,c}$ : the desirability of assigning Marine  $m$  to class  $c$  (note: the fitness score is zero for Marine-class pairs where either the Marine does not meet the classes mandatory properties, or where the class does not fulfill the Marine's PEF guarantee.)

$demand_c$ : demand for Marines to be trained at class  $c$

$penalty_c$ : penalty for each unit of demand not met (higher value means it is more critical to fill the class)

- Decision Variables

$x_{m,c}$  (binary integer): 1, if Marine is assigned to the given class; 0 otherwise

## 3. Objective

Maximize the Total Utility: (Total Reward - Total Penalty)

$$TU = k_1 \left( \sum_{c \in \mathcal{C}} \sum_{m \in \mathcal{M}} fit_{m,c} \cdot x_{m,c} \right) - k_2 \left( \sum_{c \in \mathcal{C}} penalty_c \cdot (demand_c - \sum_{m \in \mathcal{M}} x_{m,c}) \right) \quad (\text{III.1})$$

## 4. Constraints

- assignmentLimit: a Marine is assignable to at most one school

$$\sum_{c \in \mathcal{C}} x_{m,c} \leq 1 \quad \forall m \quad (\text{III.2})$$

- eligibility: a Marine is only assignable to a class they are fit for (this prevents assigning Marines with a fitness score of zero)

$$x_{m,c} \leq fit_{m,c} \quad \forall m \quad \forall c \quad (\text{III.3})$$

- capacity: since penalties apply only if there is a positive shortfall, the objective function would be non-linear. To avoid that, we assume we will never oversupply Marines to schools, resulting in:

$$\sum_{m \in \mathcal{M}} x_{m,c} \leq demand_c \quad \forall c \quad (\text{III.4})$$





## IV. A DECISION SUPPORT SYSTEM FOR RECRUIT DISTRIBUTION

### A. ARCHITECTURE AND IMPLEMENTATION

Following is a discussion of the RDdss architecture and its implementation. First, the architectural components are discussed. Then, the steps taken to build the RDdss are described.

#### 1. Architecture

The architecture of the RDdss is depicted in Figure 7. This illustration shows how the six major components of the system are related. We will describe this architecture by examining each component in the following order: switchboard, relational database, preprocessor, assignment model, solver, and analyzer.

The switchboard is one of two components providing an interface between the user and the RDdss. It is the mechanism by which the user controls the operation of the system. Figure 8 illustrates the main switchboard of the RDdss. It is displayed

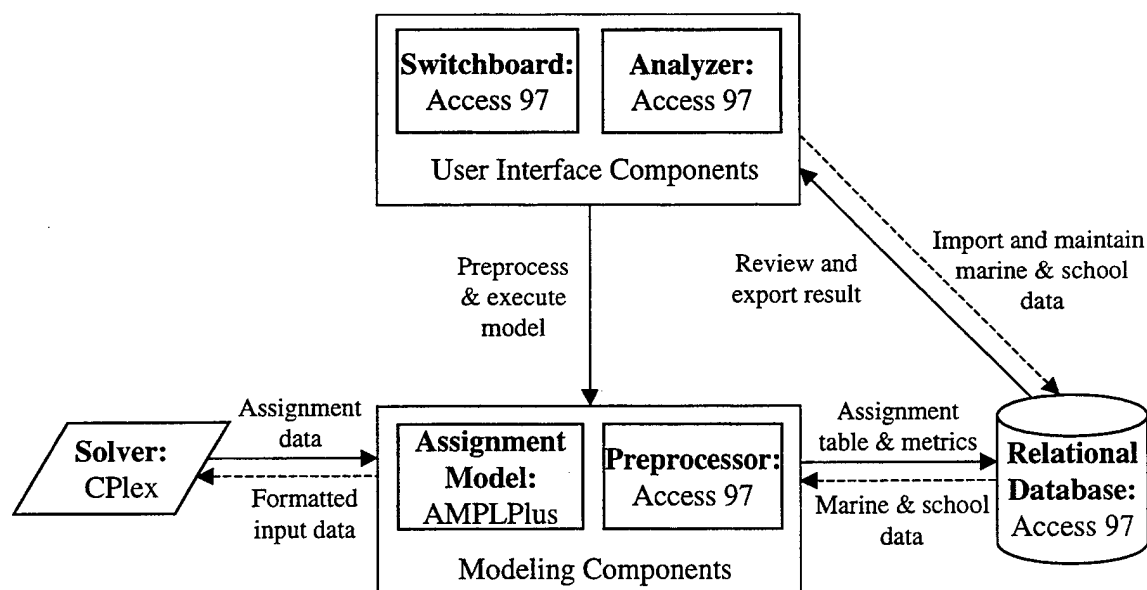


Figure 7. RDdss Architecture

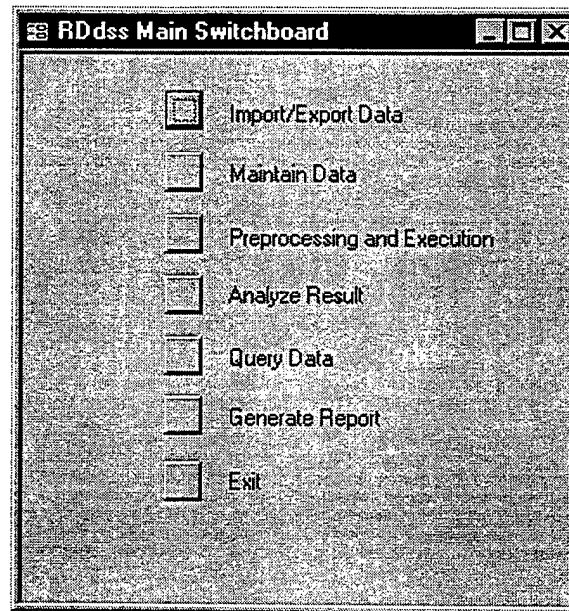


Figure 8: RDdss Main Switchboard

when the application is started. As this illustration depicts, the user can perform the following functions by simply pointing and clicking on the appropriate button: import and export data, maintain data, preprocessing and execution of the assignment model, analyze results, query data, and generate reports. All the RDdss switchboards were generated using Access 97.

The relational database technology was used in the RDdss [Ref. 8]. The actual table relationships are shown in Figure 9. These tables were developed in Access 97, which uses the relational technology. In addition to a few flat files used to communicate with the assignment model, all the data for the RDdss is stored in this database.

The preprocessor is one of the two modeling components. It consists of Access 97 Visual Basic for Applications (VBA) code that computes information necessary for the assignment model [Ref. 9]. It determines the demand of each class, the appropriate penalty associated with each class, and the Marine-to-class fitness matrix. The code for these assignment model inputs is found in Appendix D, lines 452-516 and 4305-4812.

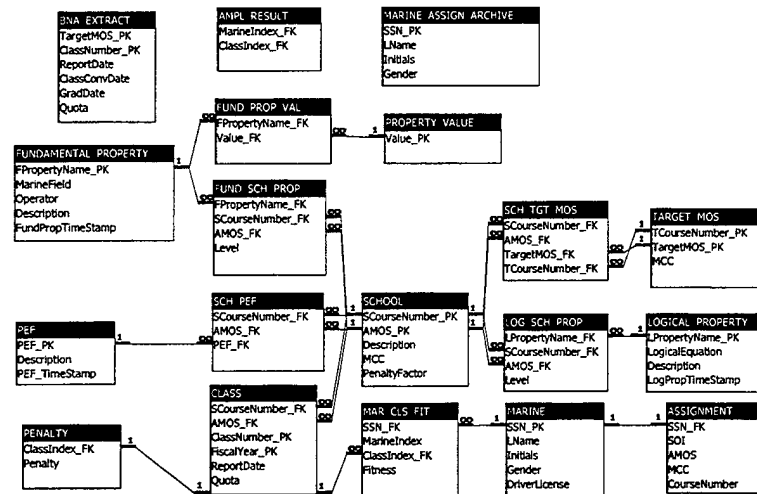


Figure 9. RDdss Relational Schema

The assignment model is the other component of the modeling components. It has been modeled in AMPL [Ref. 10], a language and a computing environment for expressing, solving, and analyzing mathematical programming problems (minimizing or maximizing a function of decision variables, subject to constraints on the variables). A complete discussion of the assignment model's math model is available in Chapter III, Section D.

The next component is the solver. It communicates directly with AMPL, the modeling language of AMPLPlus. As mentioned in Chapter II, it uses an algorithm called CPLEX, which is an optimization package for linear, network and integer programming. Working in conjunction with AMPLPlus, the solver finds an optimal solution.

The final component is the analyzer. It is the other component making up the user interface. This is where the RDdss manager analyzes assignment results, seeking insights for developing a "good" solution. Insight is developed mostly by the graphs and numerical information available in the analyzer. These were generated in Access 97 using its report generator and VBA code. The programming used in the analyzer is shown in Appendix D and includes lines 1-310, 2156-2214, and 3756-3827.

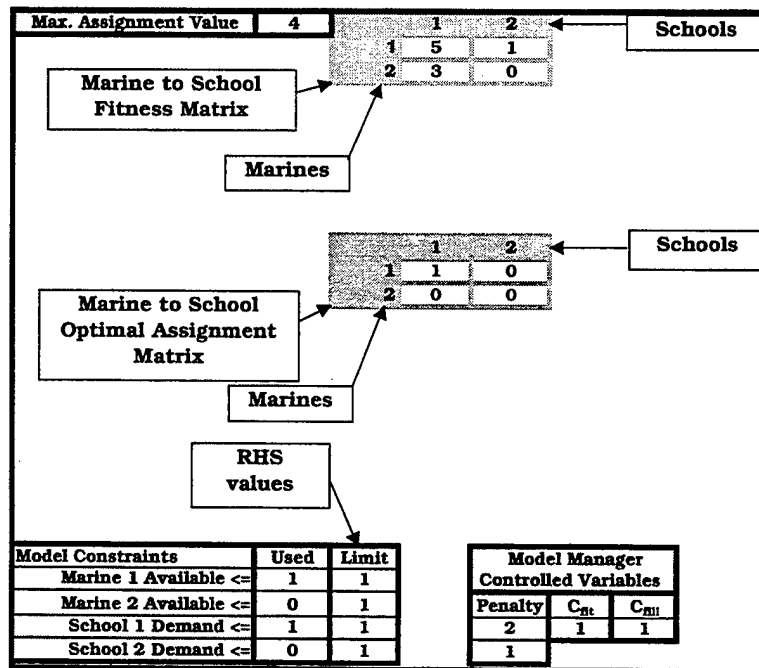


Figure 10. RDdss Demonstration Solver

## 2. Implementation

Building the RDdss was an iterative process. Some steps took numerous iterations, and some were straightforward. Below is a listing of the 14 major steps it took to build the RDdss.

1. Analyzing the current system was the first step. This included 1) interviewing USMC and DSAI personnel, 2) reviewing available documentation, and 3) operating the RDM.
2. After gathering the current system data, it was necessary to develop a logical understanding of the system. This was accomplished by modeling it with BPWin. Appendix B is the result of this step.
3. With this system understanding, it was possible to envision how the new system would make assignments. Using standard optimization techniques [Ref. 11], a small-scale demonstration solver was built using Excel 97. It is shown in Figure 10.
4. Combining the system understanding and the data requirements for the demonstration solver, it was possible to envision a data structure for the reengineered system. This led to the development of a third normal form relational schema for the RDdss [Ref. 3].

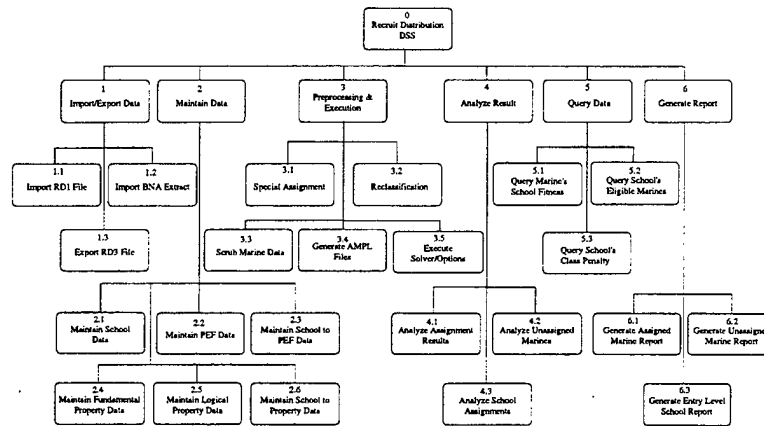


Figure 11. RDDss Decomposition Diagram

5. Using the relational schema as a model, the RDDss tables were built in Access 97. The results of this were seen earlier in Figure 9.
6. As the vision for the reengineered system came more into focus, an intuitive navigational scheme was conceived. This led to the development of a decomposition diagram for the RDDss. It is demonstrated in Figure 11.
7. Using the decomposition diagram as a model, the user interface was built for the RDDss in Access 97. The Main Switchboard was seen earlier in Figure 8.
8. With the switchboards now in place, VBA coding was commenced [Ref. 9]. This made each switchboard functional and useful.
9. Building upon the concept of the demonstration solver from step 3, a formal mathematical assignment model for the RDDss was developed. This was introduced in Chapter III, Section D.
10. The COTS applications, AMPLPlus and CPLEX were installed on the personal computer containing the RDDss.
11. With AMPLPlus now installed, it was possible to code the assignment model into the application. This made it possible to verify our design.
12. Manually creating input files for AMPLPlus is time consuming. So, the preprocessor was coded. This automated the generation of the class demand, class penalty, and Marine-to-class fitness matrix.
13. Further automation of the RDDss was accomplished by interfacing Access 97 and AMPLPlus. This required both VBA and AMPL coding. The end result was a seamless operation of the two applications.

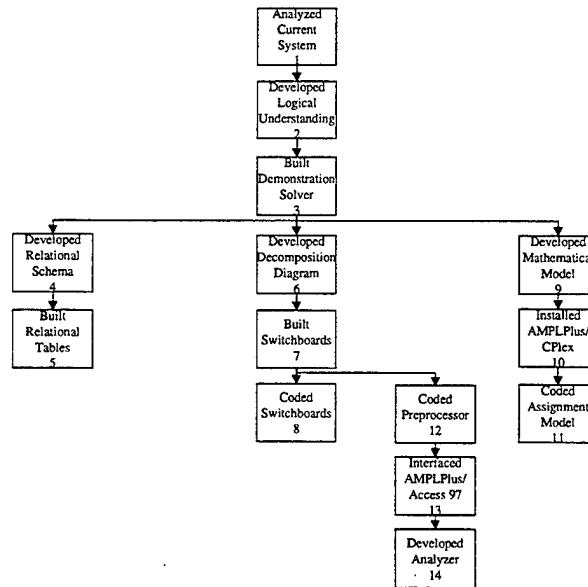


Figure 12. Actual RDdss Build Sequence

14. Finally, much thought was given to ways of gaining insight into the assignment results. This led to the development of the analyzer, which was discussed briefly in the previous section.

Some of these steps were performed in parallel. The actual order followed in building the RDdss are depicted in Figure 12.

## B. USING THE RECRUIT DISTRIBUTION DECISION SUPPORT SYSTEM

In this section, we will discuss using the RDdss. Three aspects are covered, 1) setting up a run, 2) model execution, and 3) customizing a run and results. Each is discussed in the order given here.

### 1. Setting up a Run

Setting up a run of the RDdss is straightforward and easy to do. Figure 13 graphically illustrates the steps involved. Starting with step 1 on the graph and working through to step 3d(iii), we will discuss a run set-up.

The RDdss manager starts at the top of the Main Switchboard. Here, he or

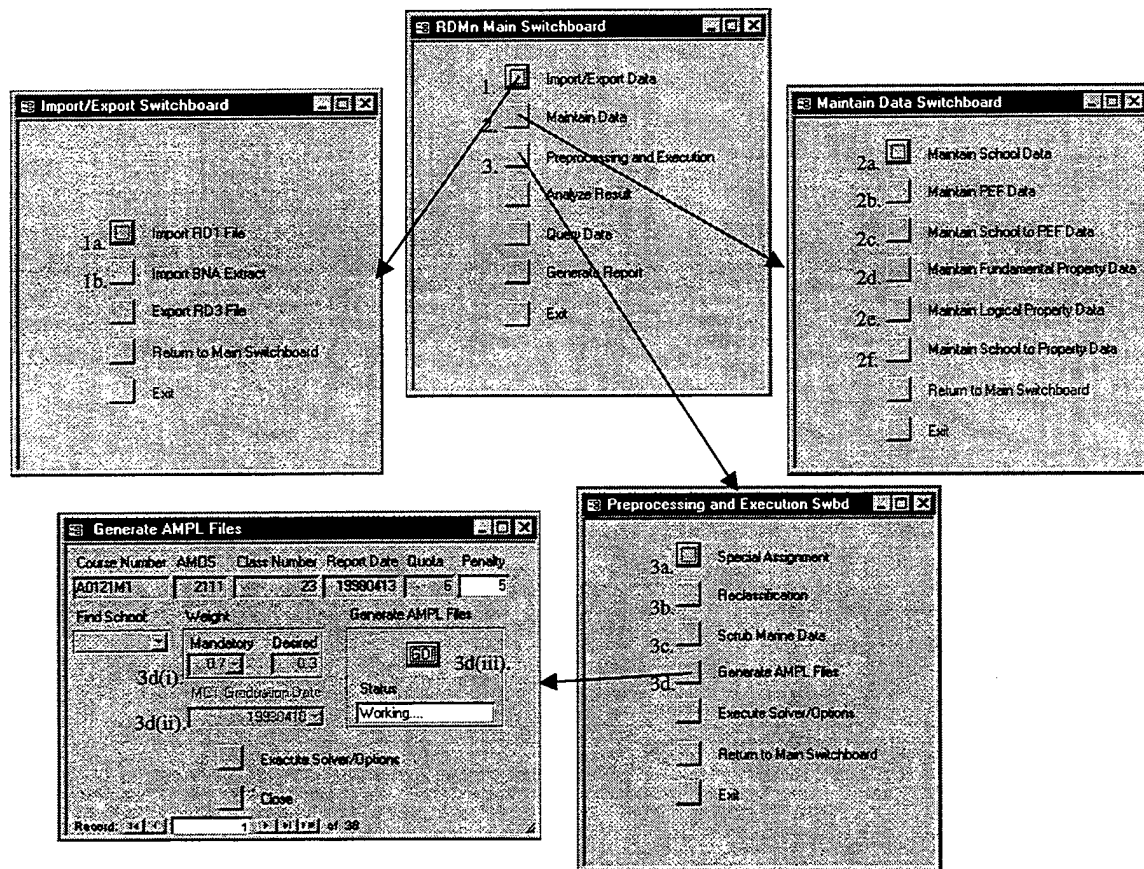


Figure 13. Setting up a Model Run

she clicks on the Import/Export Data button. This will cause the Main Switchboard to close and the Import/Export Switchboard to open. From this switchboard, the RDdss manager will import the RD1 file and the BNA extract into the RDdss MODS. This is easily accomplished by clicking on the corresponding buttons for importing these files. Finally, the RDdss manager will click on the “Return to Main Switchboard” button, causing the Import/Export Switchboard to close and the Main Switchboard to reopen.

The next step is data maintenance. This includes updating, editing, and/or deleting of school related information. The RDdss manager gets to the Maintain Data Switchboard by clicking on the appropriate button on the Main Switchboard.

From the Maintain Data Switchboard, the RDdss manager will start at the



top and work his or her way down to the bottom. The various buttons on this switchboard open forms where data maintenance is performed. For instance, the first button opens the Maintain School Data form, allowing the RDdss manager to create, edit, and/or delete a school. The other buttons allow similar operations with the PEFs, and fundamental and logical properties. After the RDdss manager has finished maintaining the data, he or she returns to the Main Switchboard by clicking on the "Return to Main Switchboard" button.

After data maintenance, the RDdss manager is ready to perform data preprocessing in preparation for a model run. By clicking on the Preprocessing and Execution button, the Main Switchboard closes and the Preprocessing and Execution Switchboard opens. From here, as on all other switchboards, the RDdss manager starts at the top and works down.

The first three preprocessing buttons directly effect the generation of the class demand, class penalty, and Marine-to-class fitness matrix. This is why they come before the fourth button, Generate AMPL Files. Preprocessing in the correct order will reduce the amount of time spent setting up the run, by minimizing the need to generate a new class demand, class penalty, and Marine-to-class fitness matrix.

The first preprocessing button concerns special assignments. Specially assigning a recruit affects the AMPL input in two ways. First, the respective member is not included in the Marine-to-class fitness matrix, since they already have an assignment. Second, the class demand for the class the recruit is assigned to must decrease by one. Otherwise, it may become overfilled.

The second preprocessing button concerns reclassification of a Marine. Reclassified Marines affect the AMPL input in two ways. First, the class demand of the class they are no longer assigned to must increase by one. This provides an opportunity to fill the empty seat during a subsequent run. Second, the class demand of the class the Marine has been reassigned to must decrease by one. Otherwise, it may become overfilled.

The third preprocessing button concerns data scrubbing, which also affects the AMPL input. It is an attempt to identify and correct errors relating to the Marines. An example is a Marine who has been given an unknown PEF. Since it is not identified by the RDdss, the member will receive a zero fitness score for every class. This will prevent the Marine from getting an assignment.

Following the completion of special assignments, reclassifications and data scrubbing, the RDdss manager is ready to generate the AMPL input files. After going to the Generate AMPL Files form, the first concern is specifying the weights for the mandatory and desired properties. This is specified by the RDdss manager at step 3d(i). What this does is indicate how important the mandatory properties are with respect to the desired properties. For instance, if the RDdss manager decided to give a weight of 1.0 to the Mandatory selection, the Desired selection would automatically fill in with 0.0. This would indicate the desired properties are of no consequence. The default for these two selections is 0.7 for Mandatory, and 0.3 for Desired.

Next, at step 3d(ii), the RDdss manager selects the MCT graduation date desired. This is selected from the drop down list. Then, the RDdss manager pushes the "GO!" button. The system will now generate the AMPL input files, completing the run set-up.

## **2. Model Execution**

Execution of the assignment model takes place after the run set-up has been completed. Figure 14 graphically illustrates the steps taken to execute the model. A discussion of this is now given.

From the Main Switchboard, the RDdss manager points and clicks on the Preprocessing and Execution button. This takes him or her to the corresponding switchboard. From here, the Execute Solver/Options selection is made. This opens up the form where the model is executed from.

The Execute Solver/Options form provides the RDdss manager the ability to set the fit and fill coefficients for the objective function. The default values are 1 for

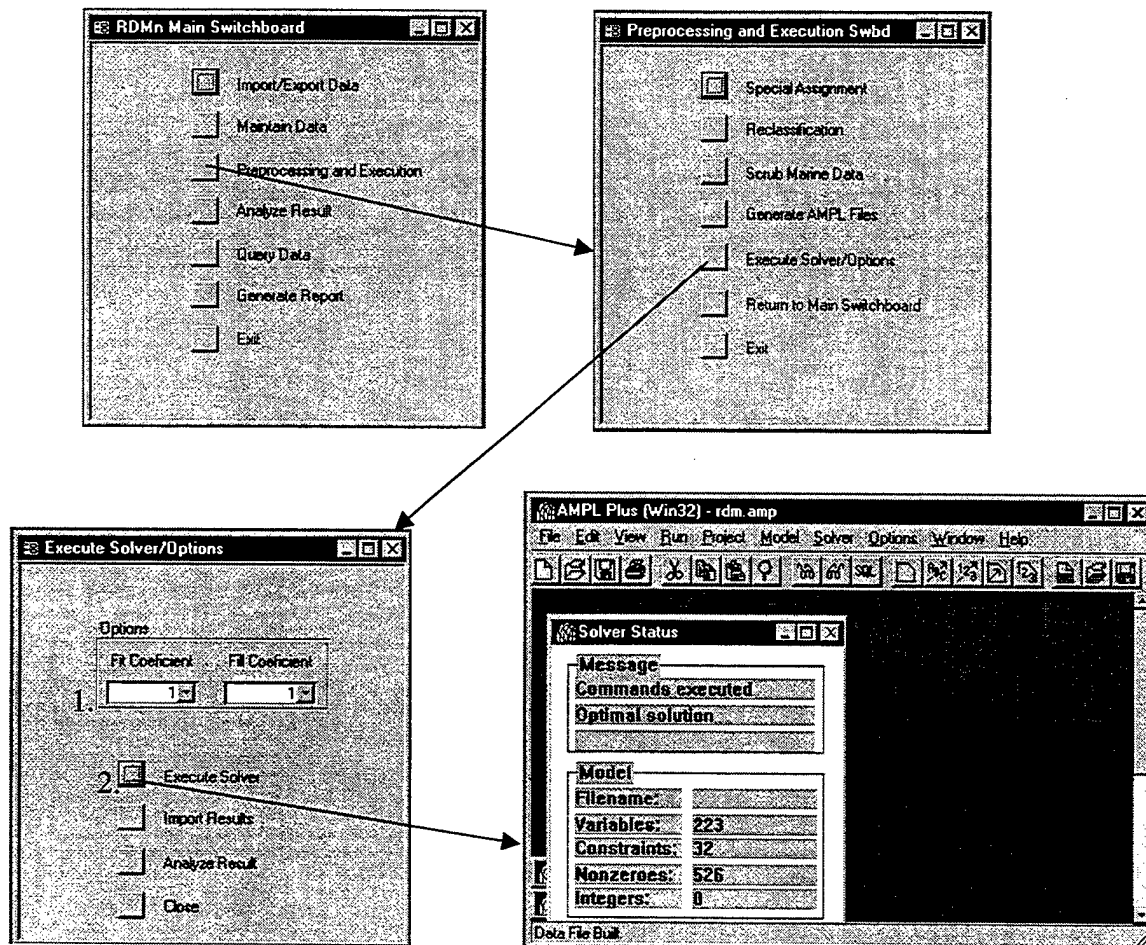


Figure 14. Model Execution

each coefficient. The RDdss manager can change either of these with a whole number from 0 - 100. The significance of these numbers is discussed in detail later in this chapter. Once the desired coefficients have been chosen, the Execute Solver button is pushed.

By selecting the Execute Solver button, AMPLPlus is launched. This application will run without user interaction. First, it will import the RDdss mathematical assignment model. Next, the specified fit and fill coefficients are retrieved. Finally, the preprocessed data generated during the run set-up are imported. Using this input data, AMPLPlus works in conjunction with the solver and generates an optimal assignment solution.

With the generation of an optimal assignment solution, the results are sent to an output file called "rdm.out." AMPLPlus, having completed its task, now closes. This indicates an optimal solution was found and is available for importing. So, the RDdss manager now imports the results by pushing the Import Results button on the Execute Solver/Option form.

### **3. Customizing a Run and Results**

As mentioned in the previous subsection, the RDdss manager can set the fit and fill coefficients of the objective function. This is a key capability of the new system. It is the means by which the RDdss manager can customize a run and "game" the RDdss. This "gaming" of the new system is essential for determining a "good" assignment result.

For instance, if the fit coefficient is set to 0, and the fill coefficient is set to a number  $> 0$ , say 1, the assignment model will maximize the fill of schools having early report dates. This has both advantages and disadvantages. One advantage is the average wait time for Marines to attend a school is minimized. Another is that the number of unfilled school seats, having a report date between the current and next scheduled model run, is minimized. However, a disadvantage is the average Marine-to-school fitness score is guaranteed to be the lowest score of all runs.

Conversely, if the fit coefficient is set to a number  $> 0$ , say 1, and the fill coefficient is set to 0, the assignment model will maximize the average Marine-to-school fitness. An advantage here is the increased chance of each Marine successfully completing their ELS, establishing a pattern of success early in their career. However, one disadvantage is the wait time for a Marine to attend a school is guaranteed to be the longest. Also, the number of unfilled school seats, having a report date between the current and next scheduled model run, is guaranteed to be the greatest.

Neither of the above solutions is ideal. A better solution is some where between these two extremes. By changing the fit and fill coefficients of the objective function, the RDdss manager has a means for customizing each run to ensure a "good"

assignment result is found.

### C. INNOVATIONS IN USER INTERACTION

A number of user interaction innovations have been made in the RDdss. Two of them have already been discussed. These were the intuitive navigational scheme using the switchboards, and the elimination of manually entering data already available in the system. One more note worthy innovation is now introduced. It is a date change innovation.

The date change innovation concerns changing the graduation date of the Marines in the downloaded RD1 file. This is necessary for generating correct assignment results. The date in the downloaded RD1 file is based on graduation from the MCRD. However, the model needs to know when the Marines are available for assignment. Availability is based on the MCT graduation date.

The old system solution to this problem took three steps. First, the RDM manager used a calendar to determine the MCT graduation date. This is normally 27 days following completion of MCRD, and is always on a friday. So, the RDM manager will find the MCRD graduation date on the calendar and count 27 days. The friday nearest to this day is the day desired. Second, this MCT graduation date, along with the RD1 file, is sent to another branch in the USMC where the date change is made. Third, the file is returned to the RDM manager, who imports the file into the RDM.

The reengineered system has streamlined this process. Within the RDdss, the RDdss manager makes the date change. It is accomplished with the push of a couple buttons.

Figure 15 shows how these changes are made. First, the RDdss manager selects the MCRD graduation date requiring a change. Once this is selected, a calendar appears. Its purpose is to help the RDdss manager correctly identify the MCT graduation date. Next, the "=" button is pushed. Its purpose is to add the number of

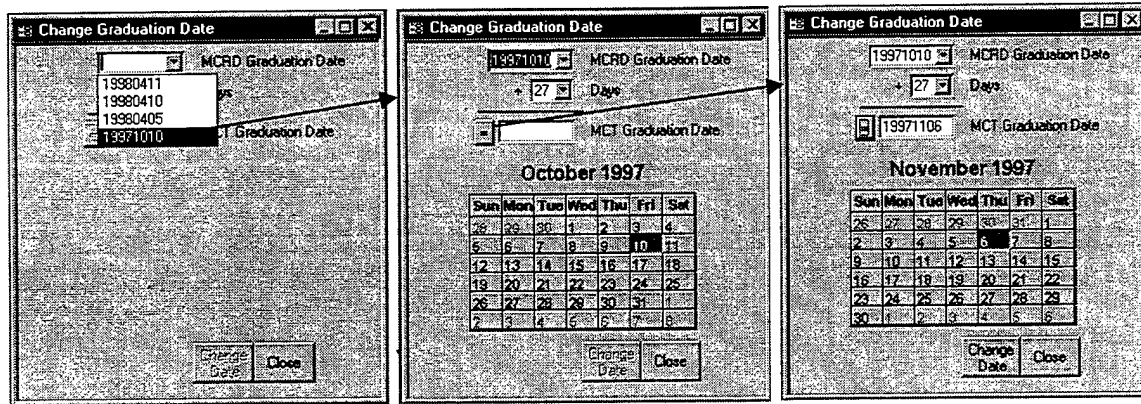


Figure 15. Changing a Graduation Date

days specified in the “Days” field (default is 27), to the MCRD graduation date. This updates the calendar as shown on the far right. When the correct date is found, the “Change Date” button is pushed. This makes the desired change to the data in the RDdss database.

## D. INNOVATIONS IN ANALYZING MODEL RESULTS

As discussed earlier, the RDdss manager has the capability of “gaming” the system by using different fit and fill coefficients. However, having this ability is not enough. The RDdss manager still needs a means to compare one run against another.

It is hard, if not impossible, to estimate the quality of a solution simply by looking at a list of assignments. The actual assignment results depend on the values of the fit and fill coefficients, which cannot be set at the ideal levels without understanding the impact on solution quality. So, analyzing run results to get insight and develop a “good” solution is now discussed.

Four measures are considered particularly useful in evaluating alternative solutions.

1. Fitness premium. The Marine-to-school fitness function is defined in a way that for each school, the average fitness over all Marines *eligible* for that school is 100. So, for any school, if the average fitness computed over Marines *assigned*

to that school is greater than (or less than) 100, the difference represents a positive (or negative) premium.

2. Unfilled seats in first week. Since the model is run nearly every week, seats left unfilled in classes starting in the first week will remain unfilled. This represents wasted resources.
3. Average wait time for Marines. Experience has shown that the longer a Marine waits before attending their ELS, the greater the chance of disciplinary problems. Therefore, it is generally better to keep this wait period short.
4. Unassigned Marines. A purpose of the system is to assign each Marine to a school. Therefore, knowing the number of unassigned Marines is an important measure.

To provide the RDdss manager insight into the assignment results, these measures have been incorporated into output representations, including summary statistics and detailed graphs. Each output provides both numerical and visual information. The numbers are used for quantitative comparisons. The graphical information provides a big picture view for use in a qualitative comparison.

As an example, consider the two fitness graphs in Figure 16. Both of these were produced using an RD1 file of 344 Marines and an BNA extract with school data covering a 120 day period. Below each graph is amplifying and numerical information. The bar chart on the left has a fit coefficient of 0 and a fill coefficient of 1. Additionally, its fitness premium is 28 ( $=128-100$ ). The bar chart on the right has a coefficient of 1 for both fit and fill. Its fitness premium is 38 ( $=138-100$ ). The fit numbers are quantitatively comparable. The fitness premium for the run on the right is 36% ( $=100\% \times [38-28]/28$ ) better than the run on the left.

The above numerical analysis is also supported by visually comparing the bar charts. Notice the large number of blips falling below the fitness score of 100 on the left graph. Each blip represents a Marine, and there are actually 44 of them. Conversely, notice the comparatively small number of blips below 100 on the right graph. There are 21 of them. The visual comparison of the two graphs reveals the

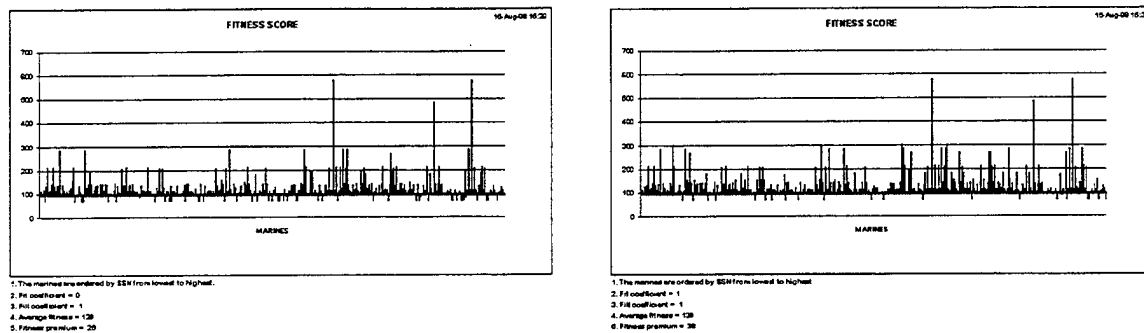


Figure 16. Two alternative results: Fit

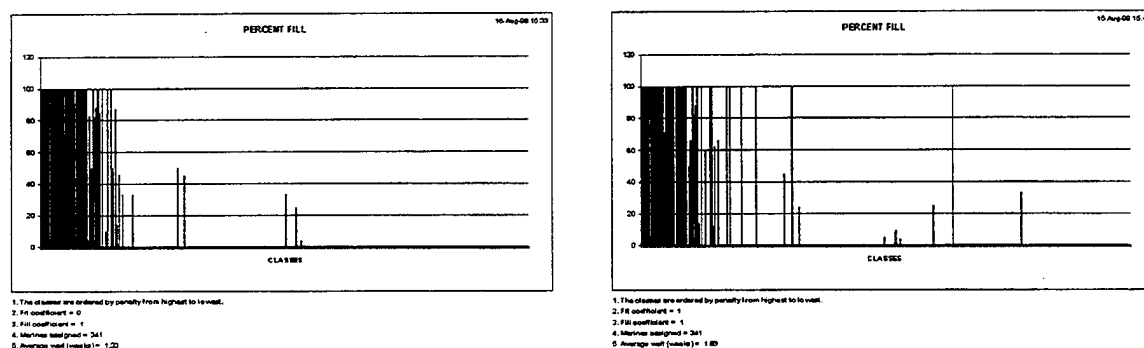


Figure 17. Two alternative results: Fill

qualitative difference. The graph on the right indicates a higher quality Marine-to-school fit. Of course, this supports the numerical comparison performed earlier.

As another example, consider the two fill graphs in Figure 17. These are based on the two model runs just described. In other words, the left graph has a fit coefficient of 0 and a fill coefficient of 1, and the right graph has a coefficient of 1 for both fit and fill.

Visually, these two graphs display an obvious difference. To appreciate how they are different requires an understanding of the significance of the blips. As note 1 of each graph indicates, "The classes are ordered by penalty from highest to lowest." Recall from Chapter III, this means the ordering is by class report date, starting with the earliest date. In other words, the bar chart on the left indicates almost every Marine will start school within the first three weeks. As discussed earlier, this is desirable. The bar chart on the right has its blips more spread out. So, by visually



comparing the two graphs, the left one has a higher quality with respect to minimizing wait time.

This is easily validated by the numbers given in note 5 of each graph. The average wait of a Marine from the left bar chart is 1.33 weeks, and the average wait from the other bar chart is 1.69 weeks. Now we quantitatively compare these values. The results from the left graph are 27% ( $=100\% \times [1.69-1.33]/1.33$ ) better than the right.

In the following table, the results from three runs are given. The measures on the left are the four measures described earlier. Each run used the same 344 Marines and 120 days worth of school classes. The information for the first and third runs is from the two model runs discussed above. The second run was determined by “gaming” the system. In other words, we tried numerous combinations of the fit and fill coefficients until a “good” solution was found. Run 2 was the result. Understand though, we are not saying “best” solution. Only the USMC can determine what is best for them.

	Run 1 (fit=0; fill=1)	Run 2 (fit=1; fill=11)	Run 3 (fit=1; fill=1)
Unfilled Seats (1st week)	23	23	23
Unassigned Marines	3	3	3
Average Wait (weeks)	1.33	1.33	1.69
Fitness Premium	28	31	38

The reason we feel Run 2 is a “good” solution is based on a quantitative comparison of the measures and one additional piece of information. It is the fact the USMC feels it is more important to minimize wait time than to maximize the Marine-to-school fit. The remainder of this section explains how we decided Run 2 was a “good” solution.

Starting with the values for unfilled seats, it is obvious all three runs are equal. The same is true for the unassigned Marines measure. So, these two measures provide no insight for comparison purposes.

Concerning the average wait, the values of Runs 1 and 2 are equal. Further, their 1.33 value is numerically smaller than the 1.69 value of Run 3. This is good for Runs 1 and 2. As was proven earlier, a value of 1.33 is 27% better than 1.69.

Finally, there is the fitness premium measure. The first observation made is that Run 1 is a candidate for elimination. The reason is as follows. Run 2 has the same score as Run 1 for the first three measures in the table, but it shows an 11% ( $=100\% \times [31-28]/28$ ) increase in the fitness premium. So, Run 1 is eliminated from the competition.

This leaves the last two runs. Run 3 shows a 23% ( $=100\% \times [38-31]/31$ ) increase in fitness premium over Run 2. This creates a conflict. Run 3 has a 27% increase in average wait time, which is bad. Conversely, it has a 23% increase in fitness premium, which is good. To decide which solution is better, more information is necessary. This comes from the fact mentioned earlier, that the USMC feels it is more important to minimize wait time than to maximize the Marine-to-school fit.

We can now conclude Run 2 is a better solution than Run 3. Here is why. Run 2 minimizes average wait by 27% by giving up a 23% increase in Marine-to-school fit. This agrees with the USMC desire to more strongly emphasize a minimum waiting time. So, Run 2 is a better solution than Run 3. Since it also satisfies the known desires of the USMC, we feel it is a "good" solution as well.

## **E. OBJECTIVE COMPARISON OF OLD AND NEW SYSTEM SOLUTIONS**

In the previous section, we described how a set of solutions generated by the RDdss are compared to determine a "good" solution. Using the same approach, we now demonstrate how to objectively compare the results from the old system with the new system. This provides the USMC a means of determining which system's solution is better, if either. Comparison of the new and old system results has been made easy by automating the importing of the RDM result into the RDdss.

We have summarized the important data for the old and new systems in the following table, based on the same 344 Marines and 120 days of school classes discussed earlier. Since the assignment algorithm for the RDM was designed to maximize fill, a run from the RDdss based strictly on fill has also been included. It is the run with a fit coefficient of 0, and a fill coefficient of 1. The last column, labeled "Run 2," is a "good" solution we found by gaming the RDdss.

	Run 1 (fit=0; fill=1)	RDM	Run 2 (fit=1; fill=10)
Unfilled Seats (1st week)	23	23	23
Unassigned Marines	3	3	3
Average Wait (weeks)	1.33	1.34	1.34
Fitness Premium	28	29	32

We start by objectively comparing the RDdss fill-based result with the RDM result. They are labeled "Run 1" and "RDM," respectively. As we shall prove, the RDdss has done a better job at filling early school seats than the RDM.

Of the four measures, only one provides any useful insight for this comparison. It is the average wait measure. The reason the other three do not apply is as follows. The first two measures have the same values, so they provide no insight. The last measure is not too meaningful, since Run 1 is strictly based on maximizing fill. In other words, it disregards fit because its fit coefficient is set to 0. This leaves the average wait measure.

A quantitative comparison of the average wait shows the RDdss solution is 1% ( $100\% \times [1.34 - 1.33] / 1.33$ ) better than the RDM solution. This is explainable by considering the approach used in each model. The RDM assignment algorithm maximizes the fill of early school seats by looking at each school in a sequential fashion. In contrast, the RDdss conducts a global comparison of Marines and schools. In other words, it goes through numerous iterations until it finds an optimal fill solution. This shows that the RDM assignment algorithm is only closely approximating the optimized assignment result.

Finally, we "gamed" the new system until a "good" solution was found. This is Run 2. It has a fit coefficient of 1 and a fill coefficient of 10.

The objective comparison of the RDM and Run 2 solutions is straightforward. Since the first three measures are the same for both results, they provide no insights. This leaves only the fitness premium measure. A quantitative comparison of this measure shows Run 2 has a 10% ( $100\% \times [32-29]/29$ ) better solution than RDM. This shows the benefit of "gaming" the RDdss.

## **F. SUMMARY**

This relatively long chapter covered some important points related specifically to the RDdss. The architectural components of the new system were discussed first, followed by the steps taken to actually build this DSS. Then, from an operational point of view, using the RDdss was discussed. This included setting up a run, executing the model, and customizing a run for determining a "good" solution. Next, two innovations were discussed in great detail. These included the date change innovation and the analyzing model results innovation. Finally, a means of objectively comparing the solution from the old and new systems was discussed.

We truly believe the innovations and capabilities introduced by this prototype can greatly benefit the USMC's Marine Enlisted Assignments branch. It can help the USMC to more effectively accomplish its mission as stated by the Commandant of the Marine Corp, "to put the right Marine in the right place at the right time with the right skills and quality of life."



## V. CONCLUSION

This concluding chapter has three short sections, followed by a final comment. In the first section, we reiterate the significant contributions made in the thesis. This is followed by a brief description of some lessons we learned. Then, some recommended improvements for the prototype are given. Following this last section, we comment on how this thesis work might benefit the other branches of the United States military.

### A. SIGNIFICANT CONTRIBUTIONS

Four significant contributions beneficial to the USMC were made in this thesis. These were discussed in detail in the previous three chapters. A list and summary of the contributions is given here.

- Analysis and articulation of the recruit distribution process - by interviewing USMC and DSAI personnel, reviewing available documentation, and operating the RDM we were able to articulate an understanding of the recruit distribution process using IDEF process modeling. Additional data modeling was articulated in a third normal form relational schema.
- Development of a mathematical model - by analyzing the assignment process, criteria, and constraints, USMC policies and objectives, a mathematical model for the new system was developed.
- Fully functional prototype - an intuitive and easy to understand new system that seamlessly interfaces with the old system was built. It provides an objective means of comparing assignment results of both systems, and was developed using COTS software applications.
- Elimination of the proprietary solver and associated contractor lock-in - this was accomplished by replacing the proprietary solver with two COTS applications.

### B. LESSONS LEARNED

Our original goal was to build an As-Is IDEF0 model of the RDM, about two months after starting the thesis. This worked out fine. The next goal was to take

another six weeks to develop a To-Be IDEF0 model [Ref. 1]. This did not work out, for two reasons. First, our understanding of the entire recruit distribution process was still not mature enough to build the To-Be model. Nearly half a year was spent just thinking about its many details. Second, it is nearly impossible to envision the new system without first knowing the capabilities and limitations of the application used to build the prototype. One extreme envisions the impossible, while the other barely touches the reengineered system's potential. *The lesson learned from this was that properly reengineering a process requires two things. A detailed understanding of the current process environment, and a good working knowledge of the capabilities and limitations of the application used to build the prototype.*

Getting feedback from the Marine Enlisted Assignments branch was not always easy. Correspondence normally occurred by e-mail. There was an inverse relationship between the number of questions asked and the number of answers given. As the one increased, the other decreased. *So, another lesson learned was that it is best to limit each correspondence to a question or two.*

While writing code, it is best to simultaneously document the code's purpose. Going back and trying to provide documentation at the end is too frustrating. This requires a considerable amount of discipline, but is well worth the effort. *The lesson learned here is document as you code.*

## **C. PROTOTYPE IMPROVEMENTS**

A big part of this thesis involved the building and refinement of the prototype. The following topics are areas where the RDdss could use improvements.

### **1. Speed Improvements**

#### ***a. Preprocessing Data***

Of all the steps necessary to complete a run of the RDdss, the preprocessing step is the most time consuming. For the 344 Marines and approximately 500 school classes analyzed in this thesis, it takes about 30 minutes to preprocess

this data. This is about ten times longer than the assignment model takes to find an optimal solution. As the size of the preprocessing data is doubled, so is the time needed to preprocess it. Finding a way to speed up this process is desirable. One possibility is to find a way to perform all the preprocessing in main memory, with only two accesses to the hard drive. The first access is to get the necessary Marine and school data. The second is to write the preprocessed results back to the hard drive.

#### ***b. Analyzing Result***

Compared to all other forms and switchboards in the RDdss, opening the "analyze results" forms is the most time intensive. With the 344 Marines and approximately 500 school classes, it takes just under a minute for any of these three to open. This feels like an eternity when you are in a hurry. The reason it takes so long is the creation of temporary tables, numerous calculations and comparisons, and retrieving of data from flat files. Finding a way to reduce this time is desirable. One possibility is to create another form which would contain only the four objective measures. Currently, the RDdss manager must go to two different forms for this information. Each containing information and displays in addition to the objective measures, which slow them down.

## **2. Multiple Solution Storage**

Currently, the prototype will only save the results of one solution. This means the RDdss manager must either print-out hard copies of the graphs, or manually write down each solution's objective measures for comparison with other runs. An improvement here should allow the storage of at least three solutions. This would permit the RDdss manager to easily compare each solution. He or she could then eliminate one or two undesirable solutions, while continuing to search for a "good" one.



### **3. Administration Switchboard**

To look at all the schools or properties in one display, the RDdss manager must open up the associated school or property table. A reason for doing this is system administration. For instance, if a supervisor is auditing the RDdss's property data, looking at all the data at one time is convenient. A simple way of including this feature is by adding another switchboard. The switchboard's buttons would have VBA code associated with them, causing the desired tables to open.

Any benefits derived from this thesis work were specifically aimed at the USMC. However, it does not have to end there. Like the USMC, other branches of the United States military must deal with the problem of how best to assign their service members to schools. Since all the services require their enlisted personnel to take the ASVAB test, the approach used in our prototype might prove an ideal solution. Further, our efforts may also contribute to the reengineering of other assignment models. So, this thesis work could actually benefit all branches of the United States military.

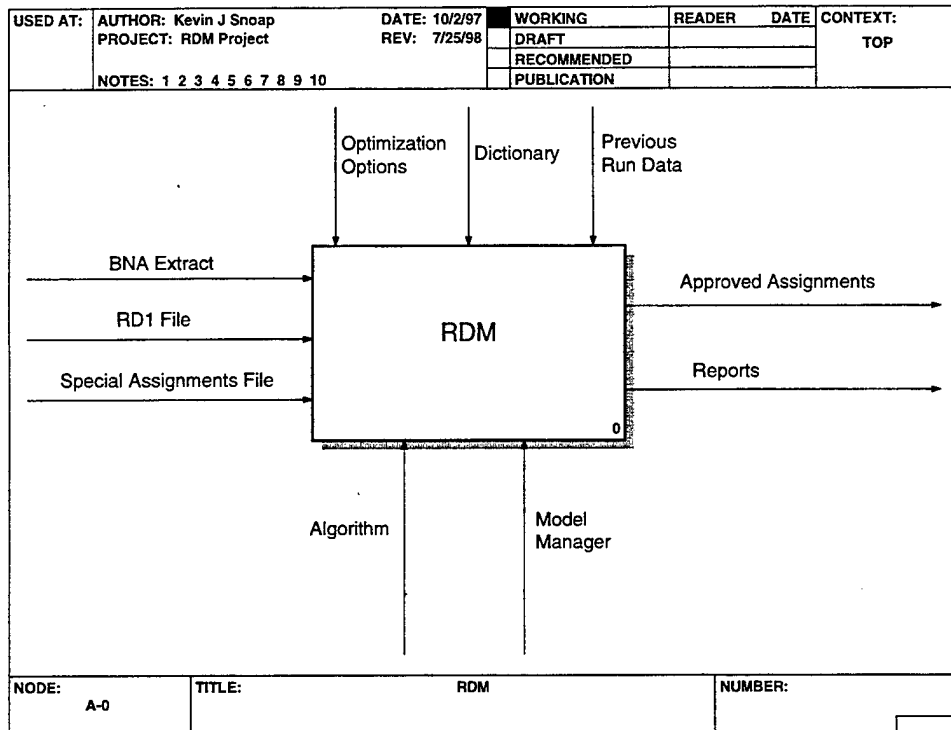
## APPENDIX A. ACRONYMS

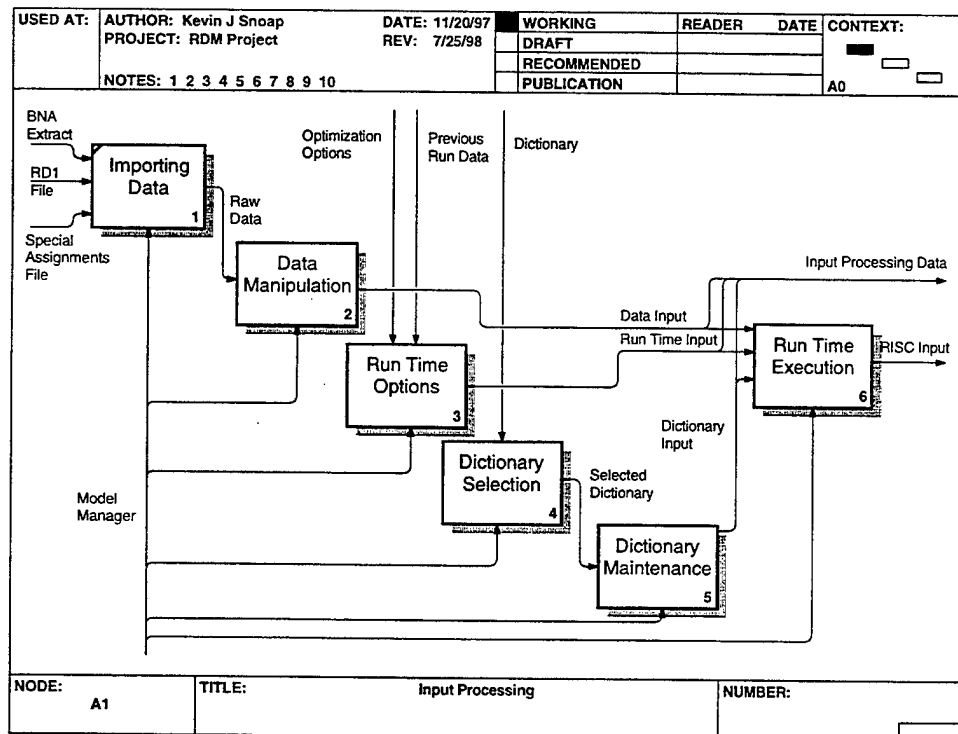
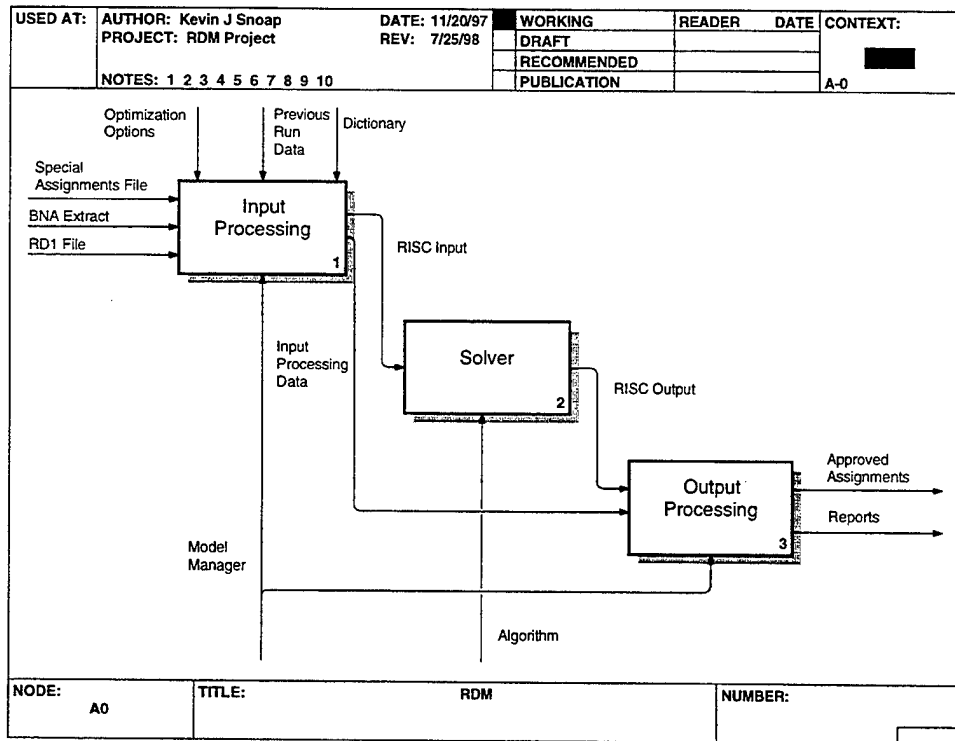
### Acronym=>Meaning

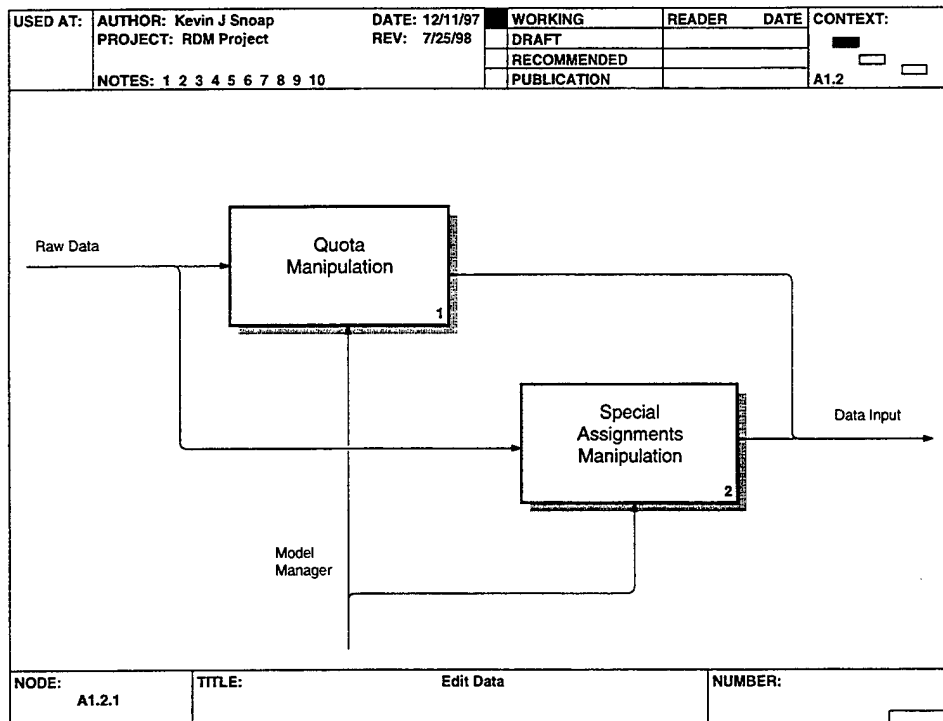
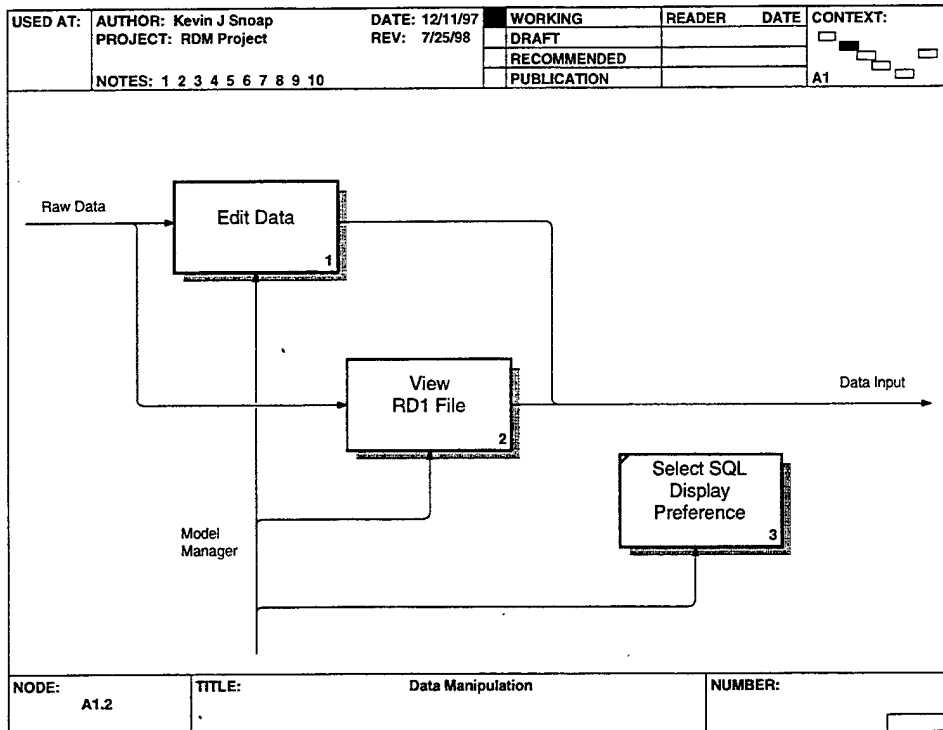
AMPL	A Modeling Language for Mathematical Programming
ARMS	Automated Recruit Management System
ASVAB	Armed Services Vocational Aptitude Battery
BNA	By Name Assignment System
BPWin	Business Process for Windows
COTS	Commercial-off-the-shelf
CPlex	Optimization Package for Complex Linear, Network and Integer Programming
DSAI	Decision Support Associates, Inc.
DSS	Decision Support System
ELS	Entry Level School
ICOM	Input, Control, Output, Mechanism
IDEF	Defense Institute Modeling
IDEF0	Business Process Modeling
MCRD	Marine Corps Recruit Depot
MCT	Marine Combat Training
MODS	Military Operational Data Store
MOS	Military Occupational Specialty
PEF	Program Enlisted For
RD1	Recruit Distribution Input File
RD3	Recruit Distribution Output File
RDdss	Recruit Distribution Decision Support System
RDM	Recruit Distribution Model
USMC	United States Marine Corps
VBA	Visual Basic for Applications

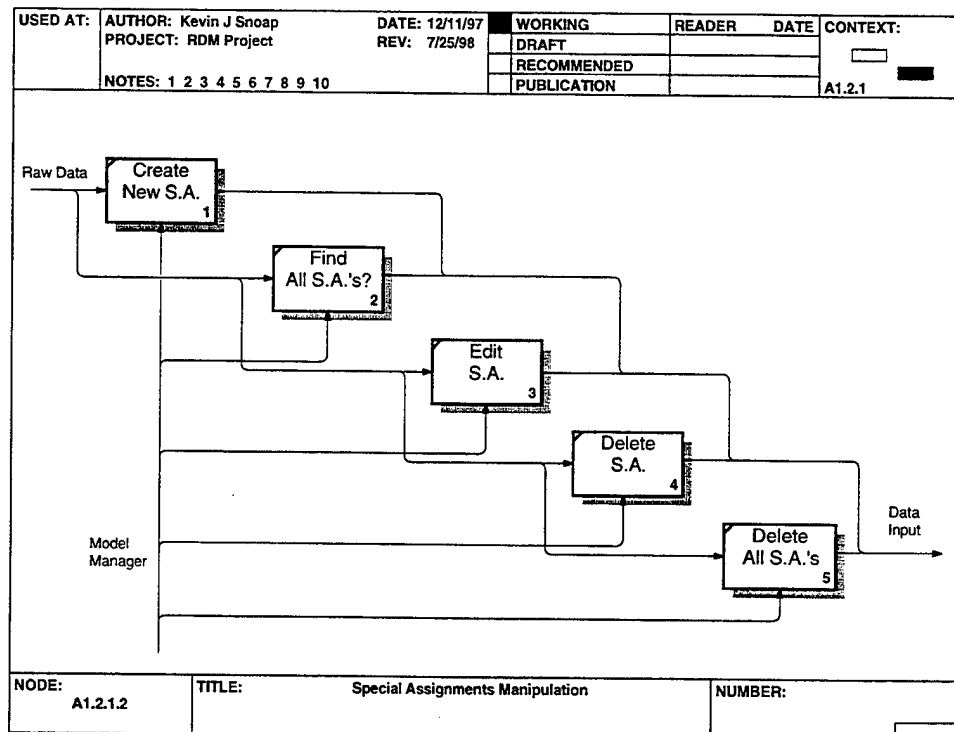
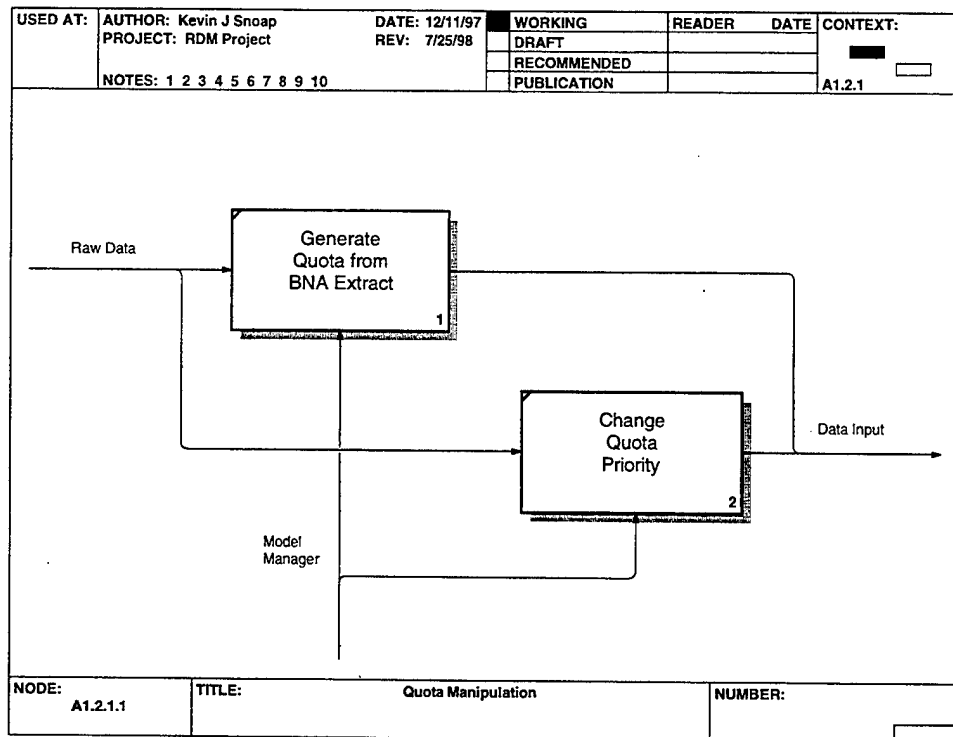


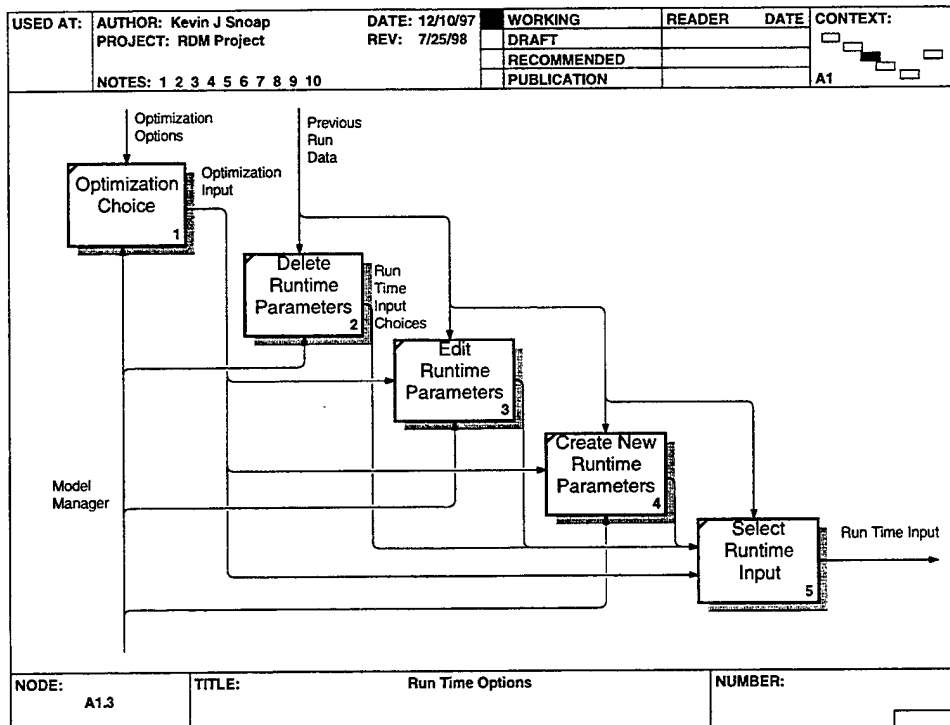
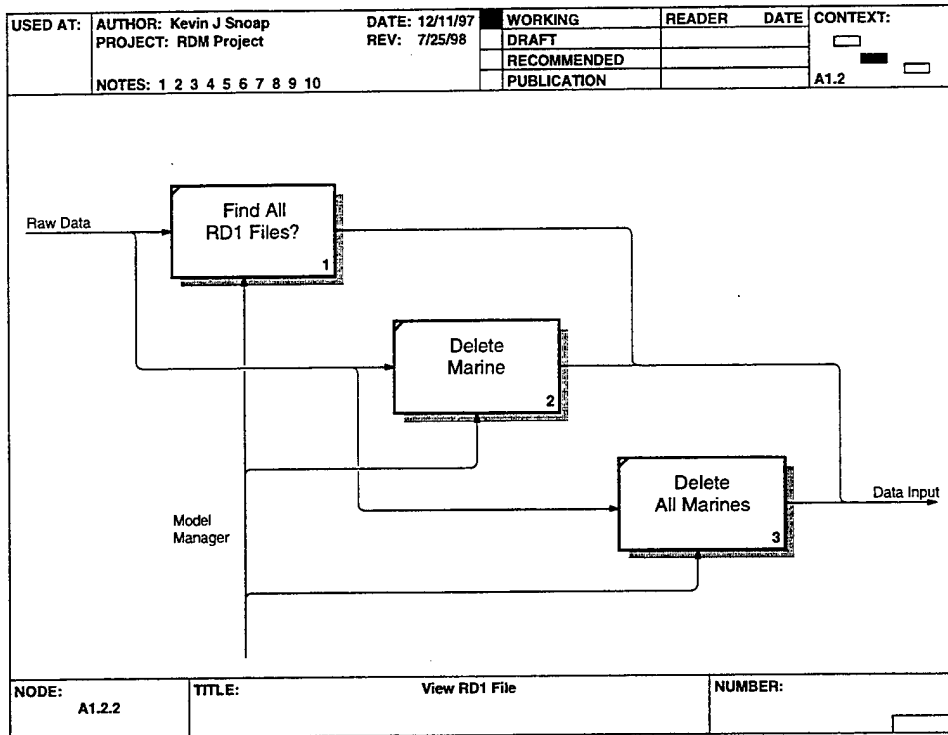
# APPENDIX B. AS-IS RD BUSINESS PROCESS (IDEF0) MODEL



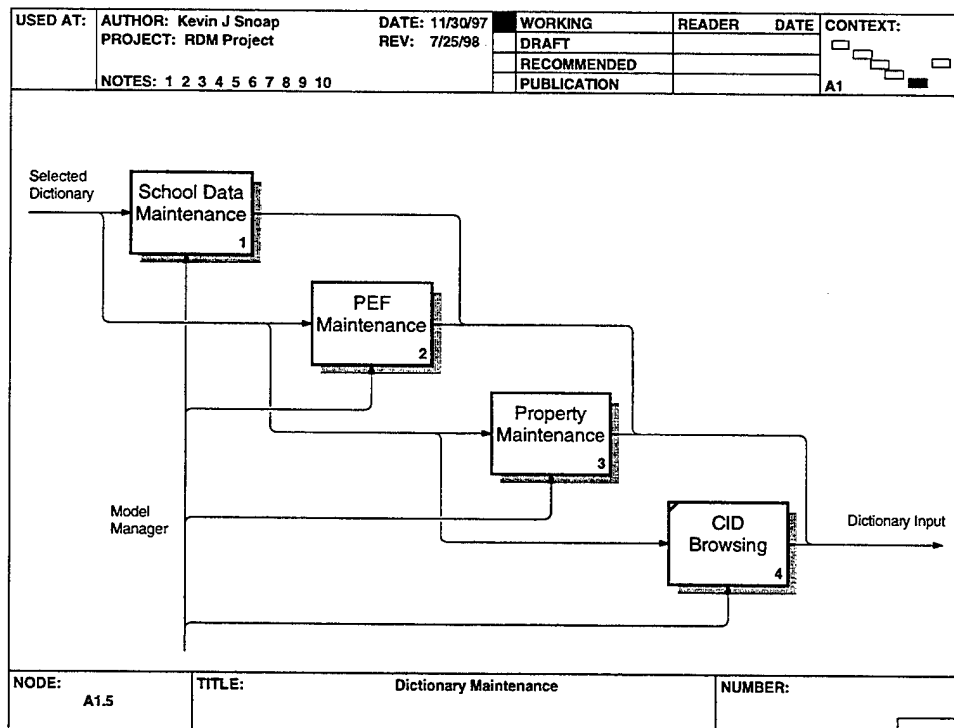
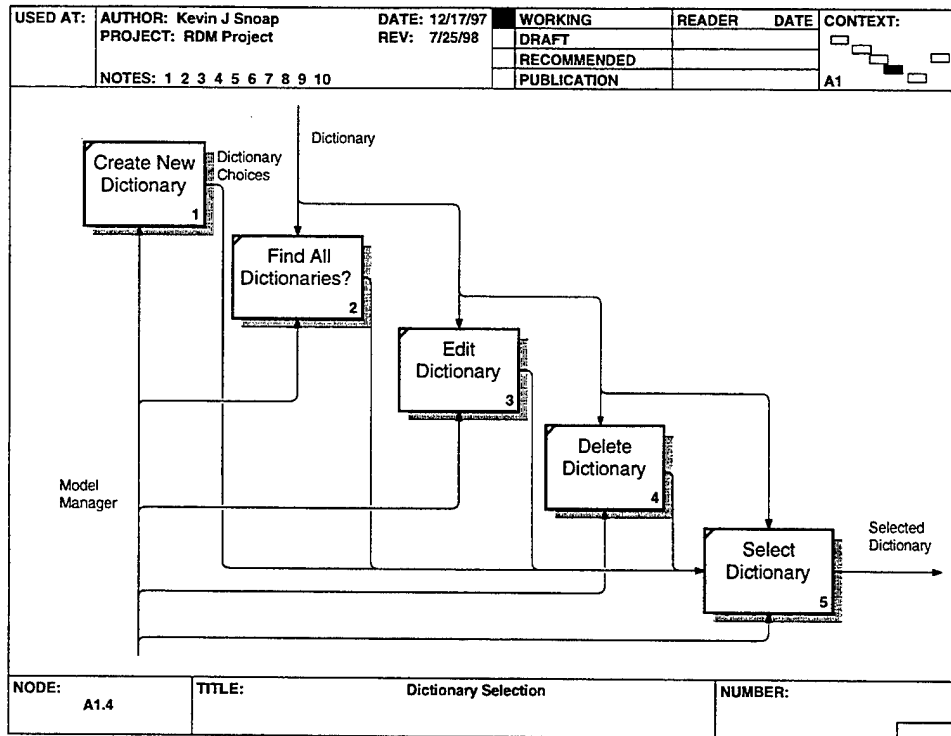


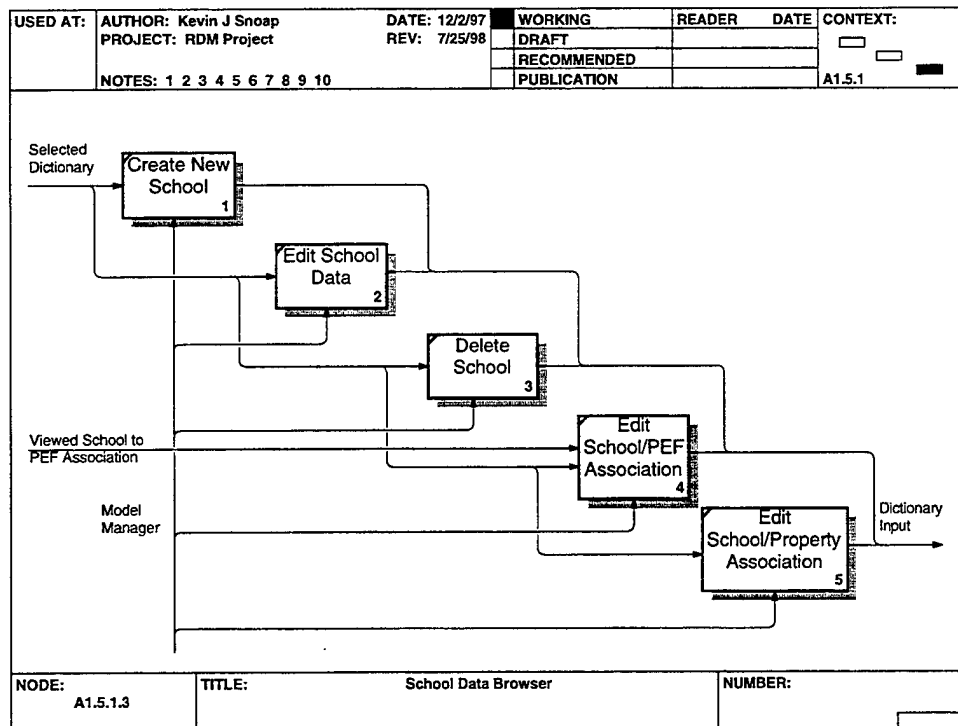
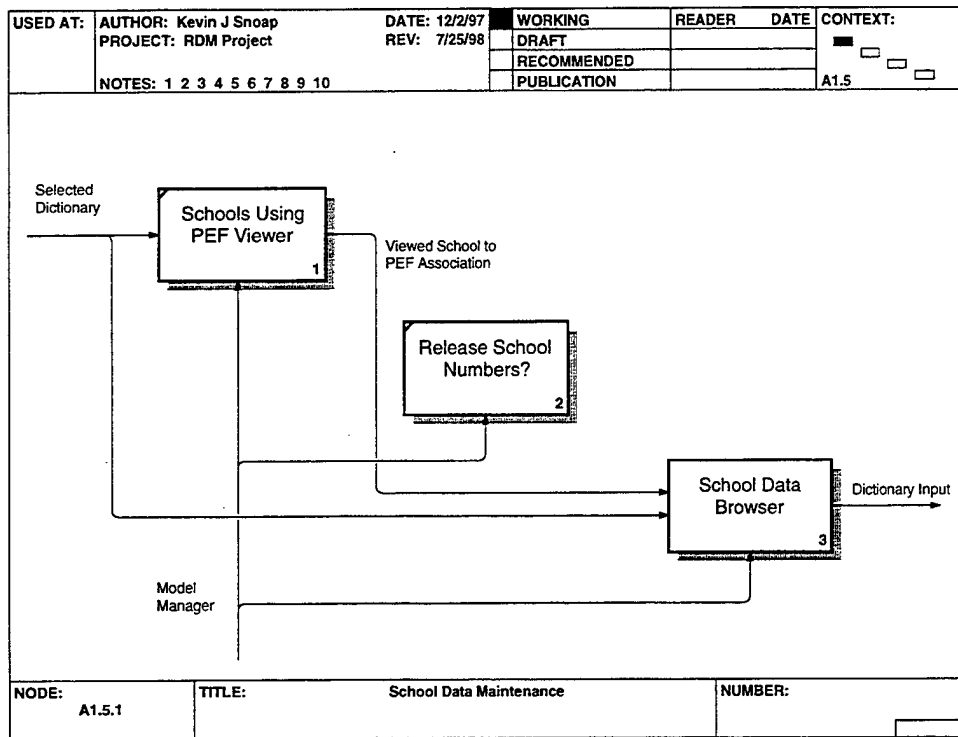


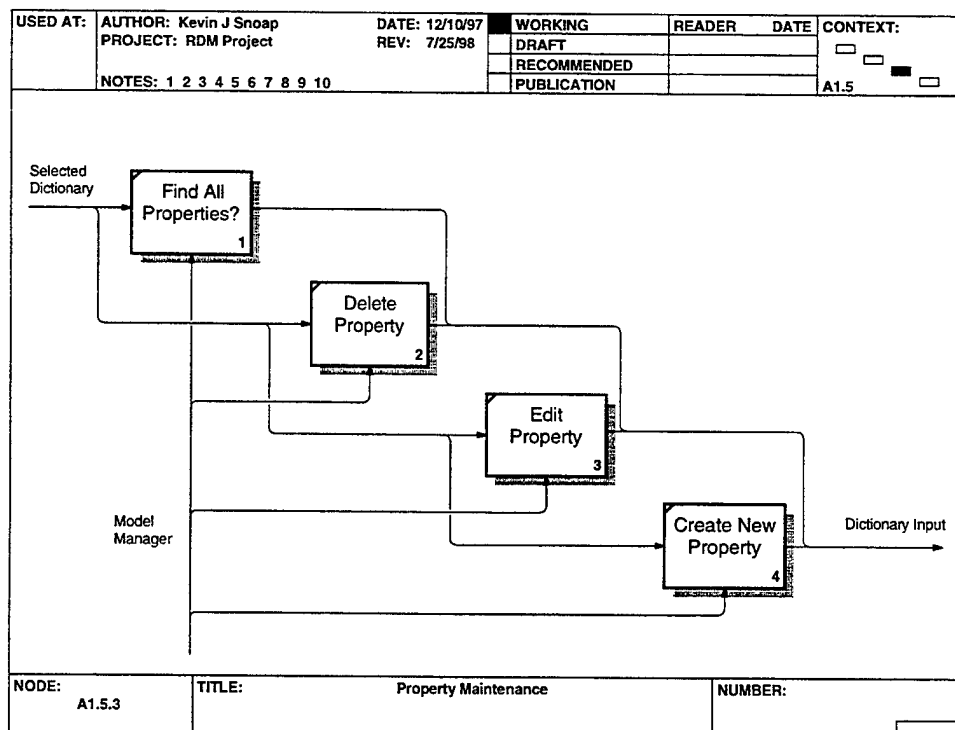
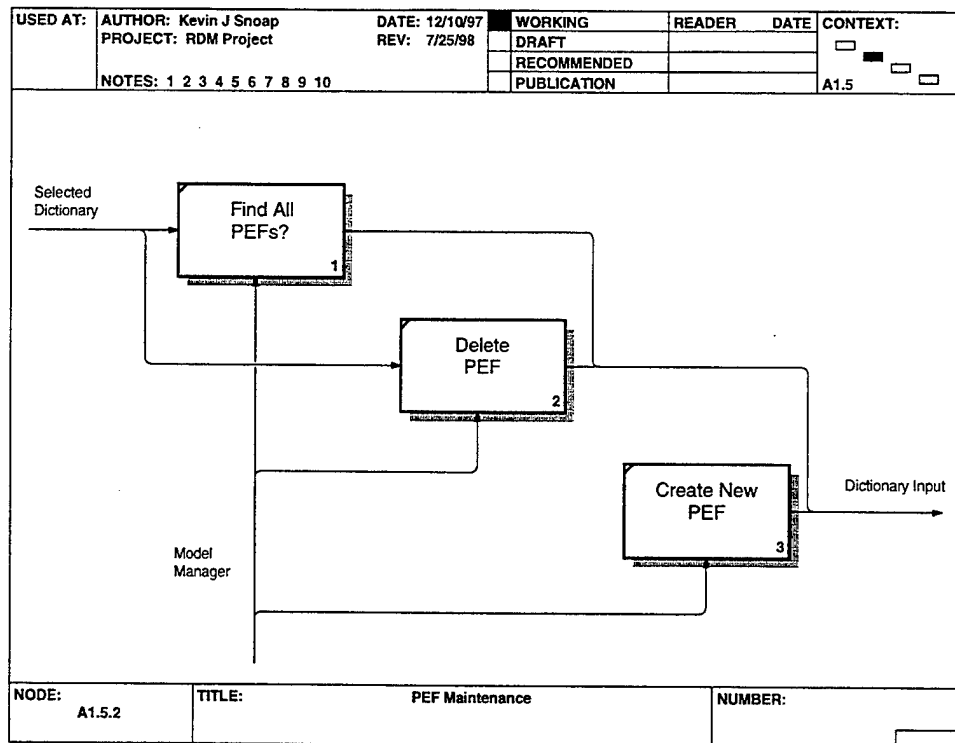


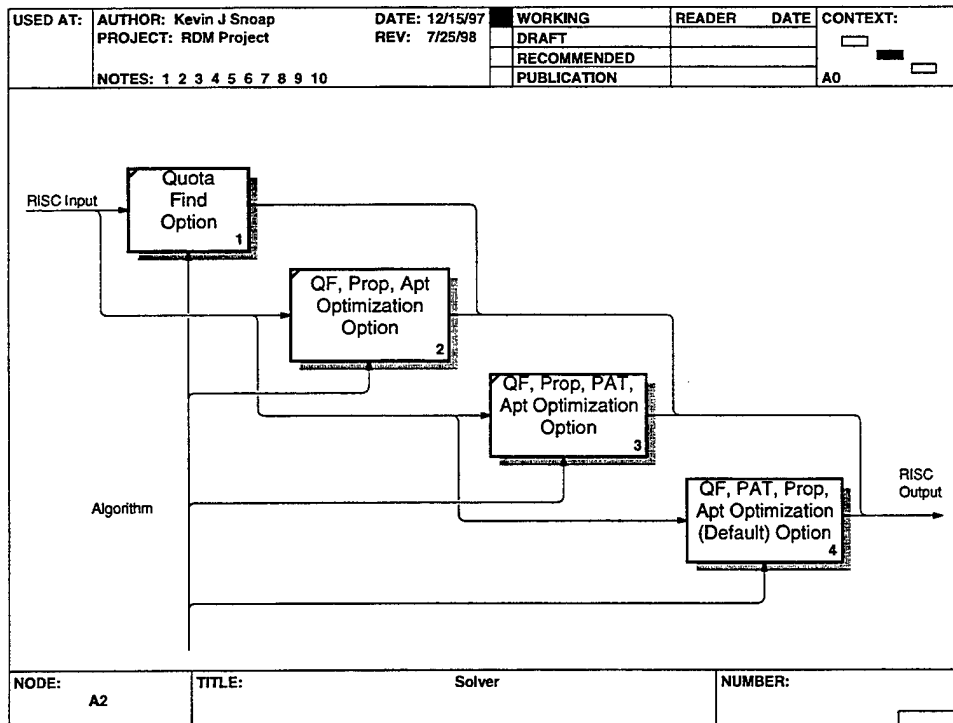
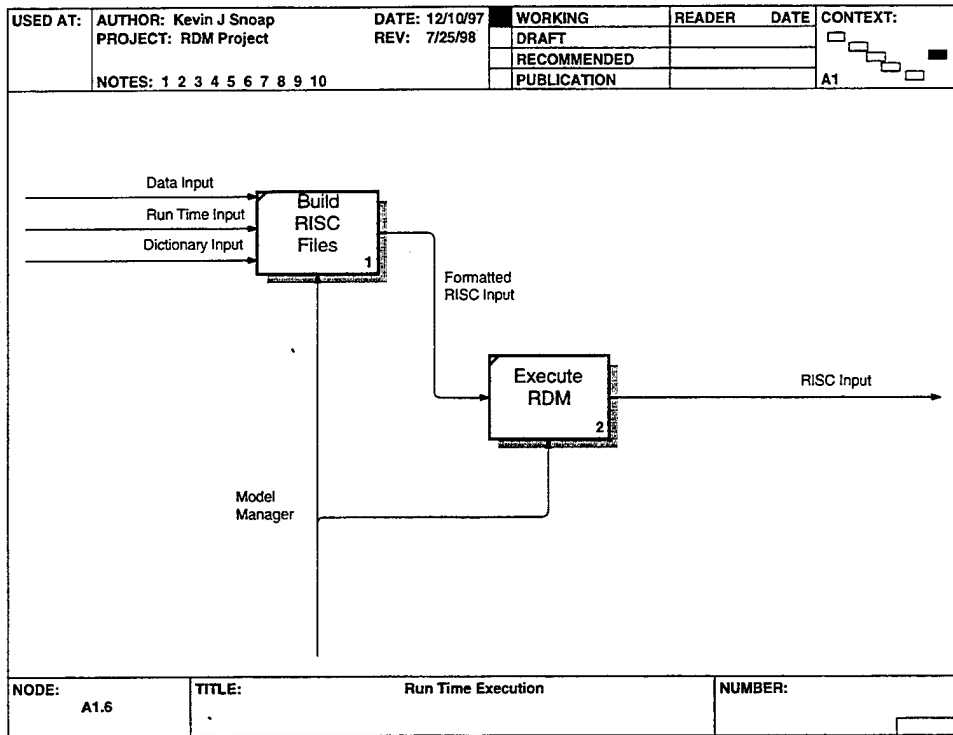


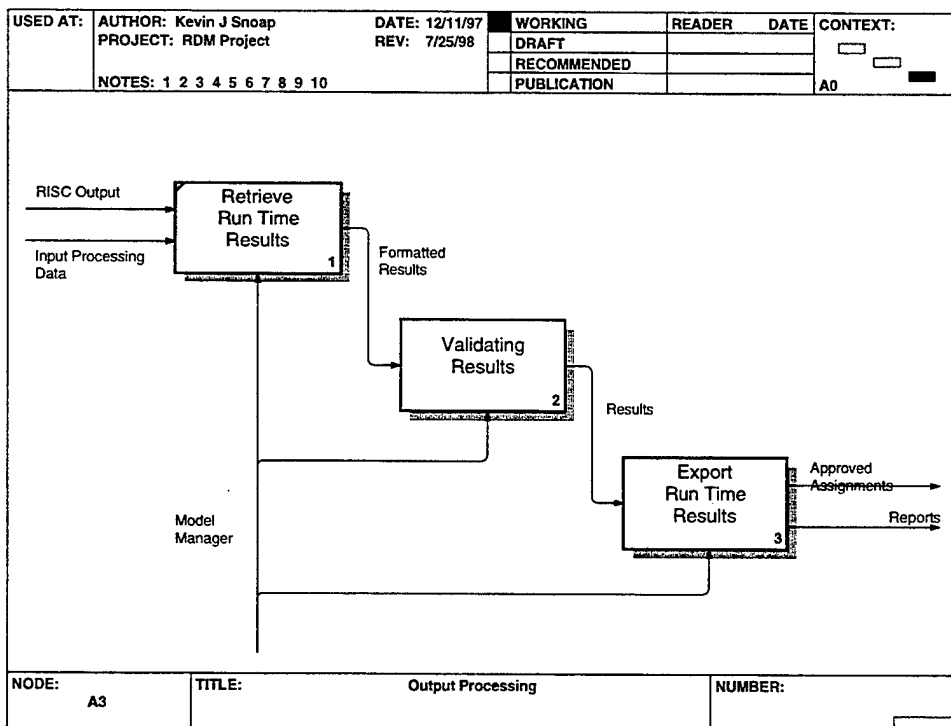
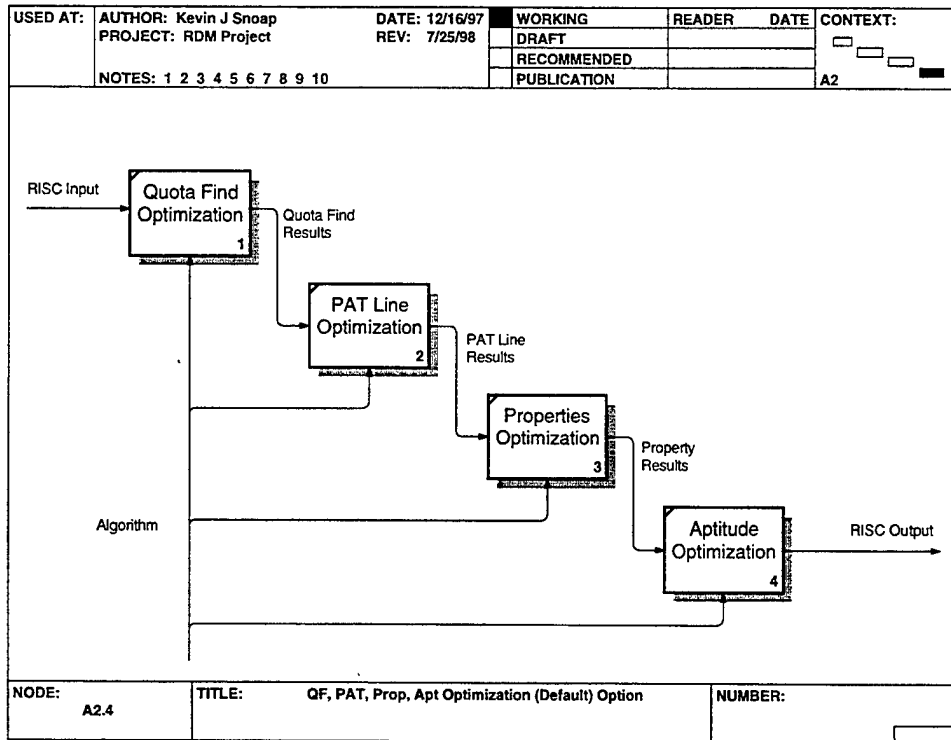


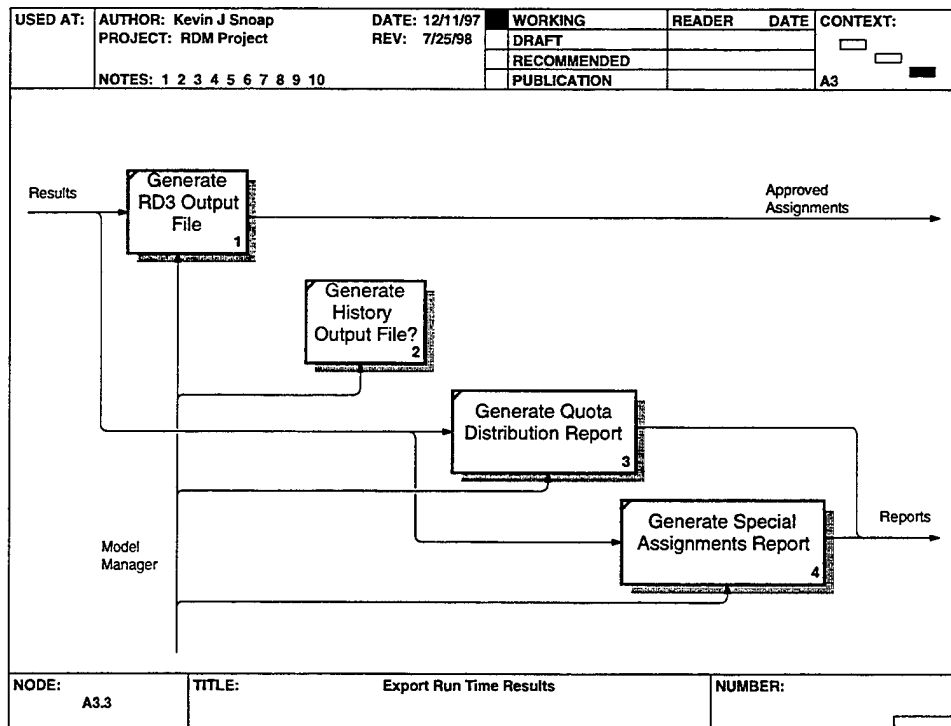
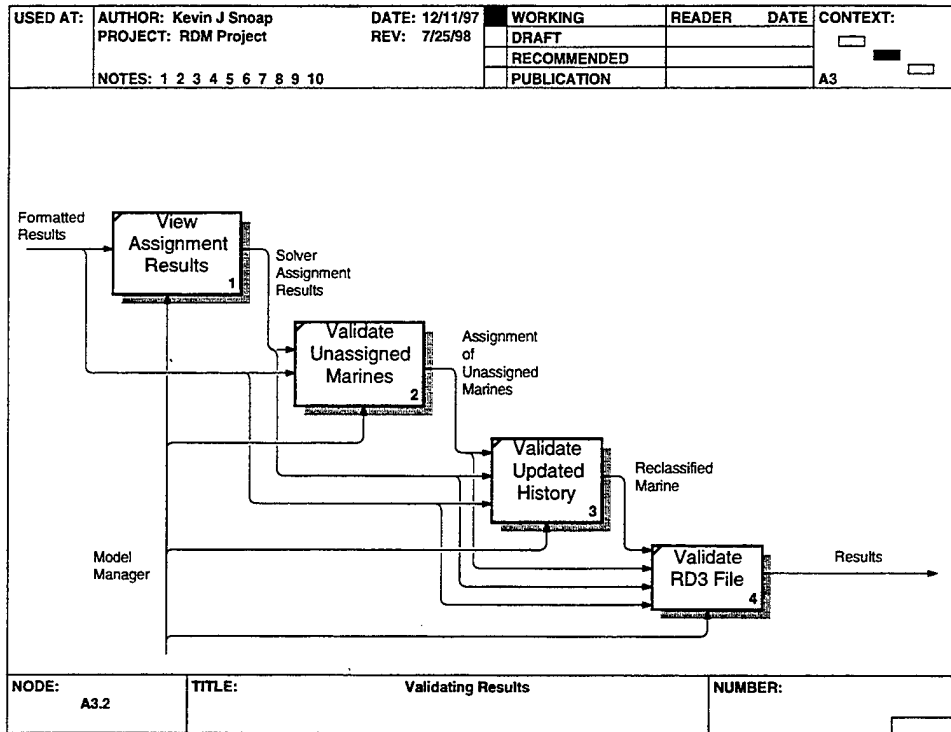






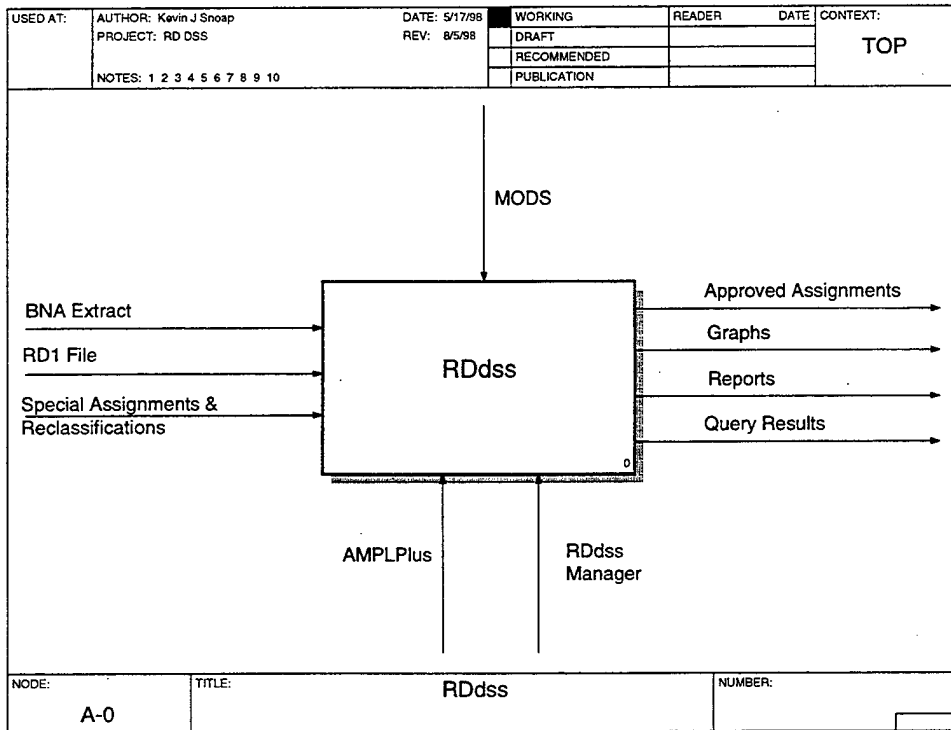




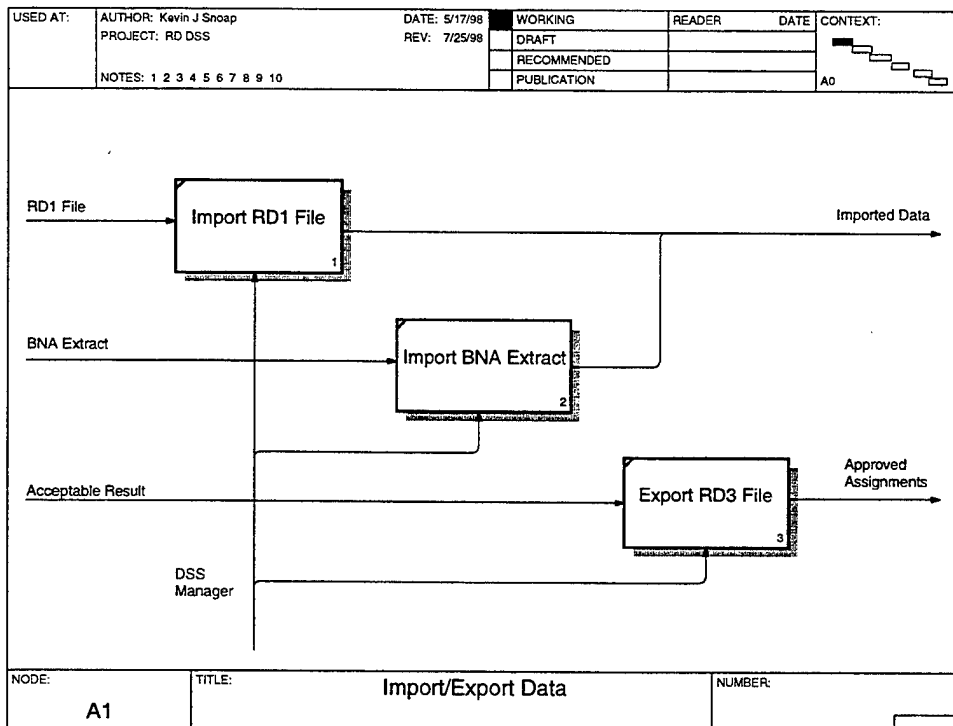
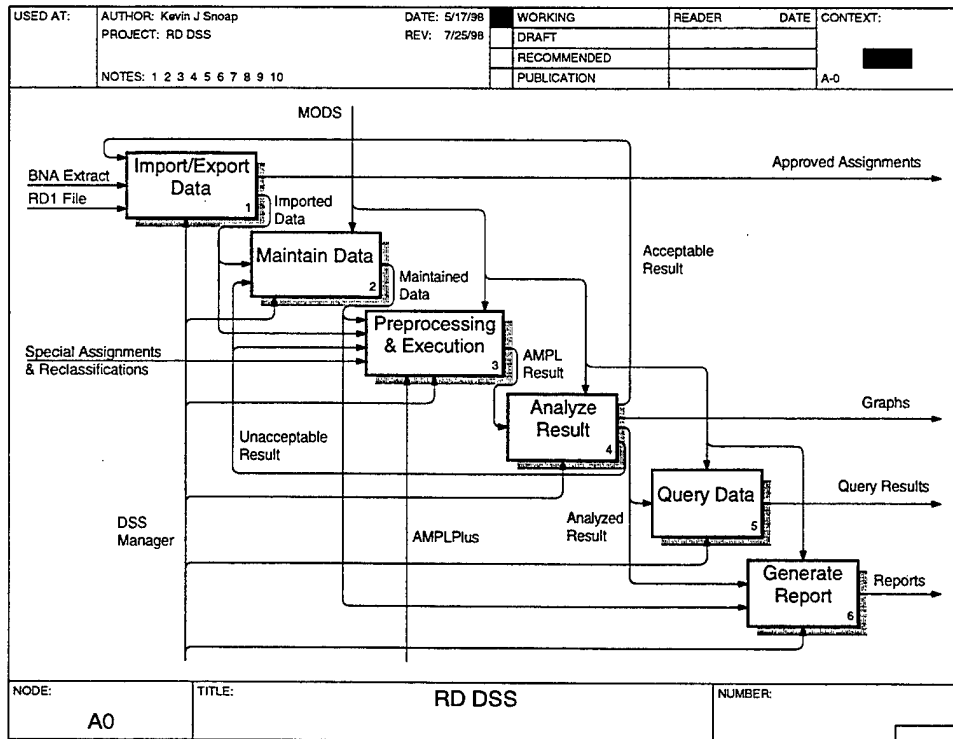


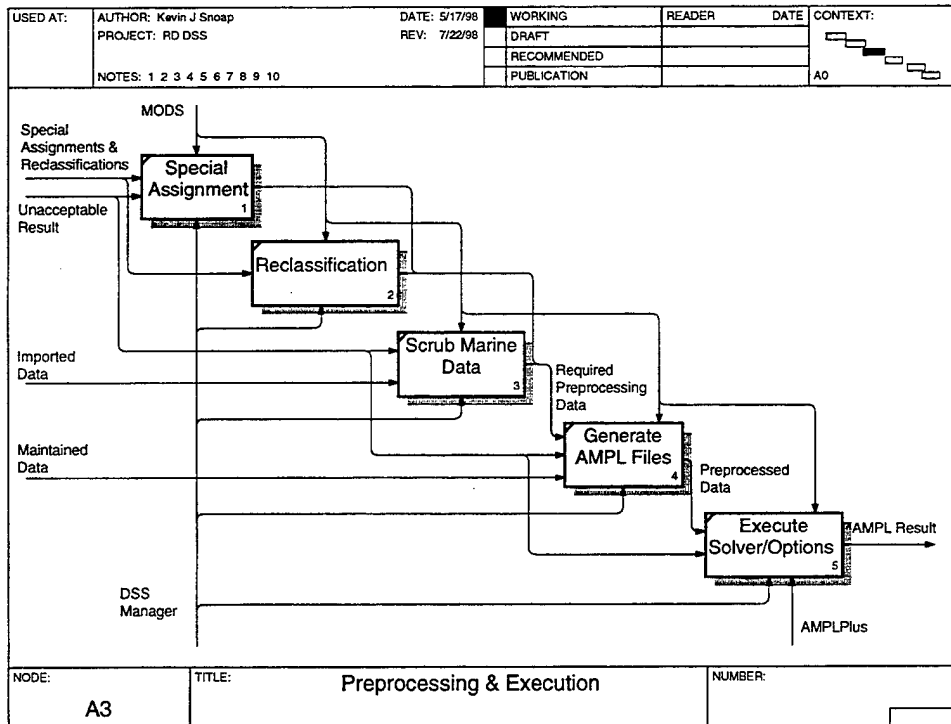
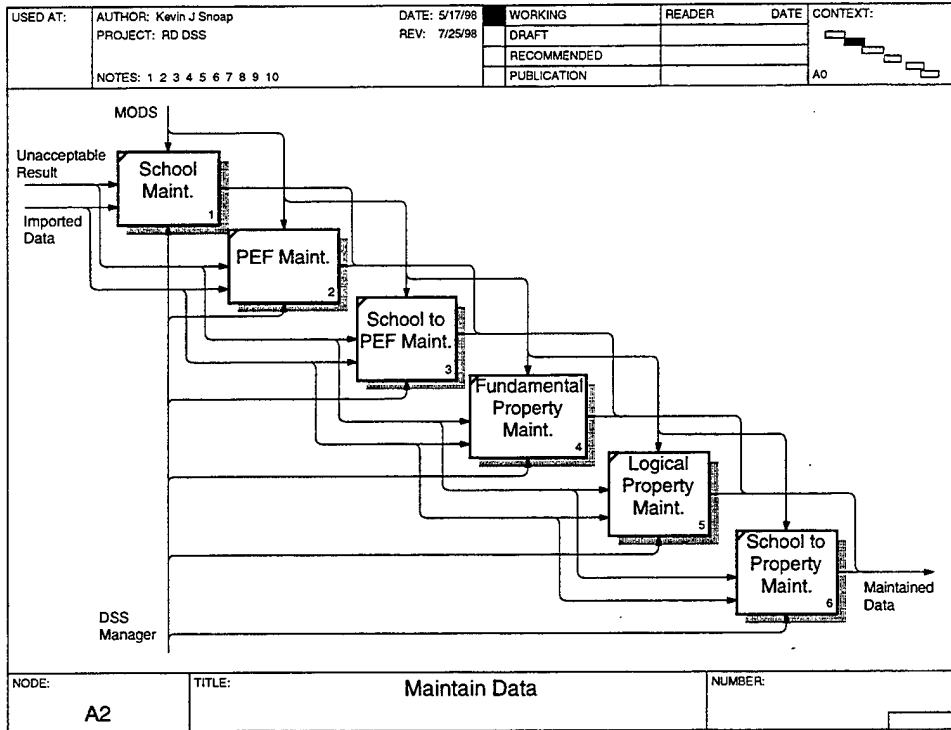


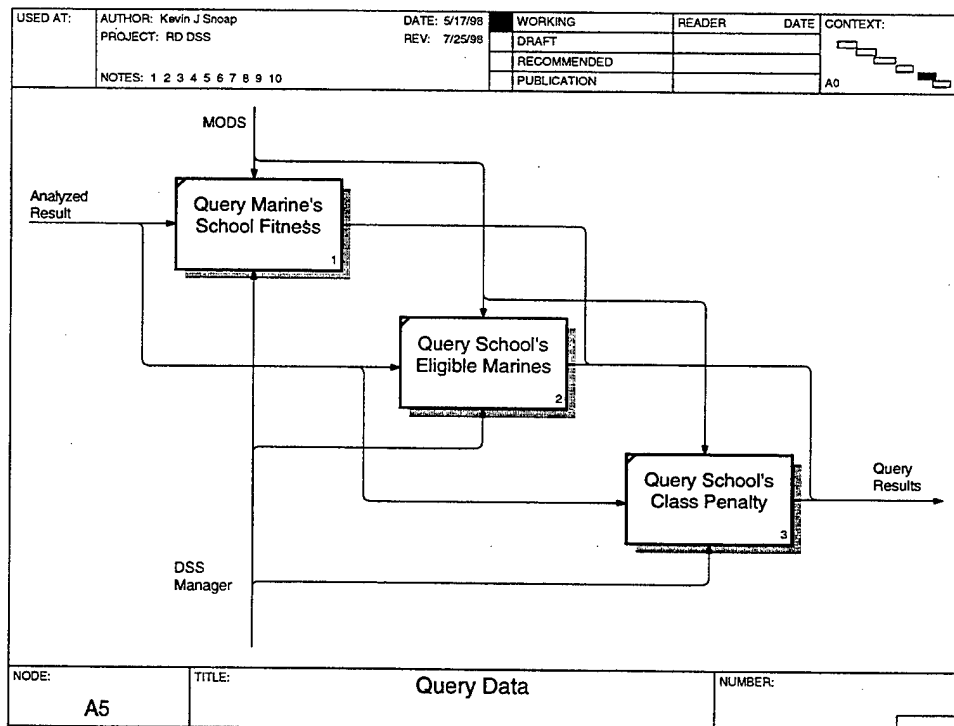
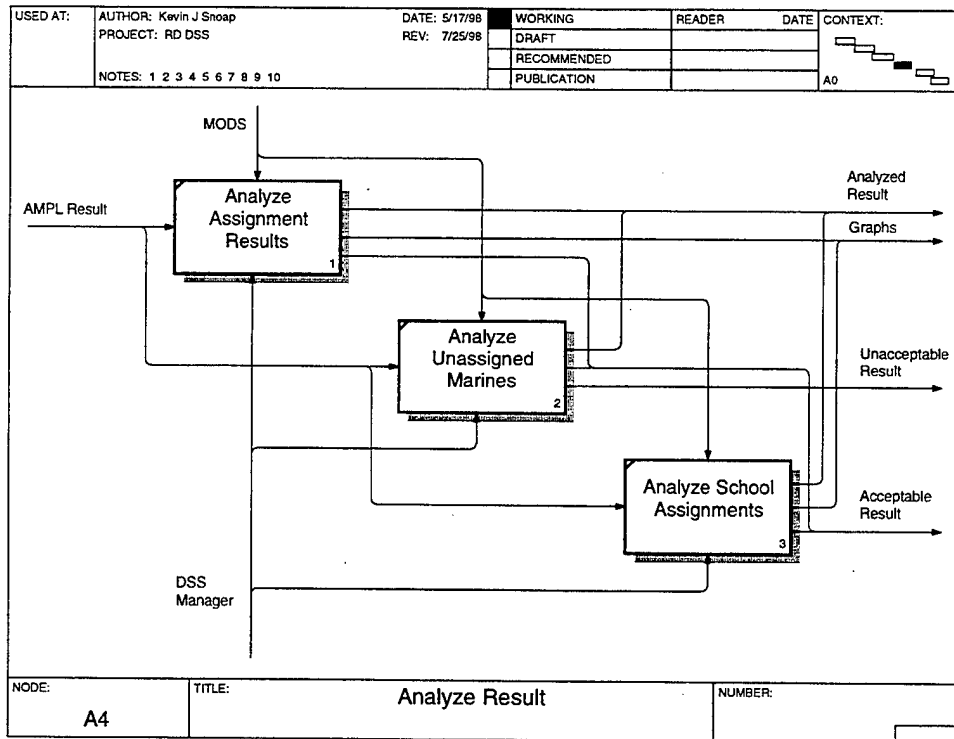
# APPENDIX C. TO-BE RD BUSINESS PROCESS (IDEF0) MODEL

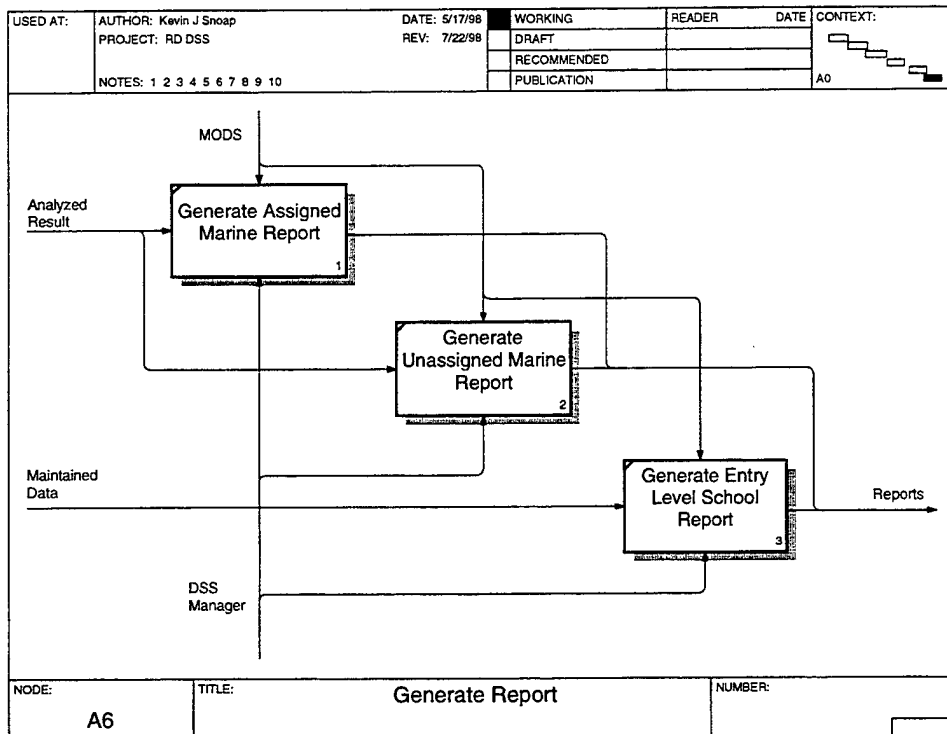














# APPENDIX D. RDDSS VBA CODE

```

1 Form: frmAnalyzeResult
2 Code
3 1 Attribute VB_Name = "Form_frmAnalyzeResult"
4 2 Attribute VB_Creatable = True
5 3 Attribute VB_PredeclaredId = True
6 4 Attribute VB_Exposed = False
7 5 Option Compare Database
8 6 Option Explicit
9 7
10 8 Dim booSchoolAssignment As Boolean
11 9
12 10
13 11 Private Sub btnExit_Click()
14 12 On Error GoTo Err_btnExit_Click
15 13
16 14
17 15 DoCmd.Close
18 16
19 17 btnExit_Click:
20 18 Exit Sub
21 19
22 20 Err_btnExit_Click:
23 21 MsgBox Err.Description
24 22 Resume Exit_btnExit_Click
25 23
26 24
27 25 End Sub
28 26
29 27 Private Sub btnSchoolAssignments_Click()
30 28 On Error GoTo Err_btnSchoolAssignments_Click
31 29
32 30 Dim stDocName As String
33 31 Dim stLinkCriteria As String
34 32
35 33 'Set the flag
36 34 booSchoolAssignment = True
37 35
38 36 'Close the current form
39 37 DoCmd.Close
40 38 'Open specified form
41 39 stDocName = "frmSchoolAssignments"
42 40 DoCmd.OpenForm stDocName, , stLinkCriteria
43 41
44 42
45 43 Exit_btnSchoolAssignments_Click:
46 44 Exit Sub
47 45
48 46 Err_btnSchoolAssignments_Click:
49 47 MsgBox Err.Description
50 48 Resume Exit_btnSchoolAssignments_Click
51 49
52 50 End Sub
53 51
54 52 Private Sub btnSolverAssignments_Click()
55 53 On Error GoTo Err_btnSolverAssignments_Click
56 54
57 55 Dim stDocName As String
58 56 Dim stLinkCriteria As String
59 57
60 58 'Close the current form
61 59 DoCmd.Close
62 60 'Open specified form
63 61 stDocName = "frmAssignmentsResult"
64 62 DoCmd.OpenForm stDocName, , stLinkCriteria
65 63
66 64 Exit_btnSolverAssignments_Click:
67 65 Exit Sub
68 66
69 67 Err_btnSolverAssignments_Click:
70 68 MsgBox Err.Description
71 69 Resume Exit_btnSolverAssignments_Click
72 70
73 71 End Sub
74 72
75 73 Private Sub btnUnassignedMarines_Click()
76 74 On Error GoTo Err_btnUnassignedMarines_Click
77 75
78 76
79 77 Dim strSQL As String
80 78 Dim rec As Recordset
81 79 Dim db1 As Database
82 80
83 81 Set db1 = CurrentDb()
84 82
85 83 'If the number of unassigned marines = 0, then disable the
86 84 unassigned marines button
87 85 strSQL = "SELECT Count(PEP) AS TotalUnassigned FROM"
88 86 Set rec = db1.OpenRecordset(strSQL, dbOpenSnapshot)
89 87 If rec1.TotalUnassigned = 0 Then
90 88 MsgBox (" All Marines were assigned.")
91 89 Else
92 90 Dim stDocName As String
93 91 Dim stLinkCriteria As String
94 92
95 93 'Close the current form
96 94 DoCmd.Close
97 95 'Open specified form
98 96 stDocName = "frmUnassignedMarines"
99 97 DoCmd.OpenForm stDocName, , stLinkCriteria
100 98
101 99 End If
102 100
103 101
104 102 Exit_btnUnassignedMarines_Click:
105 103 Exit Sub
106 104
107 105 Err_btnUnassignedMarines_Click:
108 106 MsgBox Err.Description
109 107 Resume Exit_btnUnassignedMarines_Click
110 108
111 109 End Sub
112 110
113 111 Private Sub Command1_Click()
114 112 On Error GoTo Err_Command1_Click
115 113
116 114 Dim stDocName As String
117 115 Dim stLinkCriteria As String
118 116
119 117 'Close the current form
120 118 DoCmd.Close
121 119 'Open specified form
122 120 stDocName = "frmIDM_Main_Switchboard"
123 121 DoCmd.OpenForm stDocName, , stLinkCriteria
124 122
125 123 Exit_Command1_Click:
126 124 Exit Sub
127 125
128 126 Err_Command1_Click:
129 127 MsgBox Err.Description
130 128
131 129 Resume Exit_Command1_Click
132 130
133 131 End Sub
134 132
135 133 Private Sub Form_Deactivate()
136 134
137 135 If booSchoolAssignment Then
138 136 Dim strSQL As String
139 137 Dim i As Integer, j As Integer
140 138 Dim rec As Recordset, rec1 As Recordset, rec2 As Recordset
141 139 Dim db1 As Database
142 140
143 141 Set db1 = CurrentDb()
144 142
145 143 'Create a table to get the fit and fit weights used during the
146 144 Application.Switchboard "Confirm Action Queries", False
147 145 'Delete the table if it already exists
148 146 DeleteTable
149 147
150 148 db1.Execute "CREATE TABLE PERCENTAGE (CourseNumber TEXT, AMOS
151 149 TEXT,
152 150 Quota INTEGER, Assigned INTEGER, Percentage INTEGER);"
153 151 Application.Switchboard "Confirm Action Queries", True
154 152
155 153 strSQL = "SELECT * FROM PERCENTAGE;"
156 154 Set rec = db1.OpenRecordset(strSQL, dbOpenDynaset)
157 155
158 156 strSQL = "SELECT * FROM qryTotalQuotaForRun ORDER BY"
159 157 Set rec1 = db1.OpenRecordset(strSQL, dbOpenSnapshot)
160 158 strSQL = "SELECT * FROM qryTotalQuotaForRun ORDER BY"
161 159 SCourseNumber_FK;"
162 160 'Ensure there are values in the records
163 161 If rec1.EOF = False Then
164 162 If rec1.EOF = False Then
165 163 rec1.MoveFirst
166 164 rec2.MoveLast
167 165 'Fill the PENALTY table with the quota values for the
168 166 For i = 1 To rec1.RecordCount
169 167 rec.AddNew
170 168 rec.CourseNumber = rec1SCourseNumber_FK
171 169 rec1AMOS = rec1AMOS_FK
172 170 rec1Quota = rec1SumOfQuota
173 171 rec.Update
174 172 Next i
175 173 rec1.MoveNext
176 174 rec.MoveFirst
177 175 'Update the PENALTY table with the percentage of quota
178 176 For i = 1 To rec2.RecordCount
179 177 rec2.MoveFirst
180 178 For j = 1 To rec2.RecordCount
181 179 If rec2.CourseNumber = rec2SCourseNumber_FK Then
182 180 If rec1AMOS = rec2AMOS_FK Then
183 181 rec.Edit
184 182 rec.Assigned = rec2CountOfRecallYear_FK
185 183 rec.Percentage = Int((rec.Assigned /
186 184 rec.Quota) * 100)
187 185 rec.Update
188 186 End If
189 187 End If
190 188 rec2.MoveNext
191 189 'If no match was found, then the percent is 0
192 190 If i = rec2.RecordCount Then
193 191 rec.Edit
194 192 rec.Assigned = 0
195 193 rec.Percentage = 0
196 194 rec.Update
197 195 End If
198 196 Next i
199 197 End If
200 198 End If
201 199
202 200 rec.Close
203 201 rec1.Close
204 202 rec2.Close
205 203 db1.Close
206 204
207 205 'Reset the flag
208 206 booSchoolAssignment = False
209 207 Else
210 208 'Delete the PERCENTAGE table
211 209 DeleteTable
212 210 End If
213 211 End Sub
214 212
215 213
216 214
217 215
218 216
219 217
220 218
221 219
222 220
223 221
224 222
225 Form: frmAssignmentResult
226 Code
227 1 Attribute VB_Name = "Form_frmAssignmentResult"
228 2 Attribute VB_Creatable = True
229 3 Attribute VB_PredeclaredId = True
230 4 Attribute VB_Exposed = False
231 5 Option Compare Database
232 6 Option Explicit
233 7
234 8
235 9 Private Sub Form_Current()
236 10
237 11 Dim strSQL As String
238 12 Dim rec As Recordset
239 13 Dim db1 As Database
240 14
241 15 Set db1 = CurrentDb()
242 16
243 17 'Calculate the high, average, and low fitness scores. Also, it
244 18 calculates the number of solver assignments.
245 19 strSQL = "SELECT Avg(Fitness) AS Average, Max(Fitness) AS Maximum,
246 20 Min(Fitness) AS Minimum, Count(Fitness) AS TotalAssigned FROM"
247 21 qryAssignmentResult;"
248 22 Set rec = db1.OpenRecordset(strSQL, dbOpenSnapshot)
249 23 Me.txtFitnessAvg = rec1Average
250 24 Me.txtFitnessMax = rec1Maximum
251 25 Me.txtFitnessMin = rec1Minimum
252 26 Me.txtSolverAssignments = rec1TotalAssigned
253 27
254 28 'Calculate the number of manual assignments, and the total number
255 29 of assignments.
256 30 strSQL = "SELECT Count(AssignmentType) AS TotalManualAssigned FROM"
257 31 qryManuallyAssignedMarines;"
258 32 Set rec = db1.OpenRecordset(strSQL, dbOpenSnapshot)
259 33 Me.txtManualAssignments = rec1TotalManualAssigned

```

```

261 Me.tblSokerAssignments
262 30
263 31 ' Calculates the number of unassigned marines, and the total number
264 of assignable marines.
265 32 strSQL = "SELECT Count(PERF) AS TotalUnassigned FROM
266 33 Sel rec = db1.OpenRecordset(strSQL, dbOpenSnapshot)
267 34 Me.tblTotalUnassigned = rec.TotalUnassigned
268 35 Me.tblTotalAssignable = rec.TotalUnassigned +
269 36
270 37 ' Create a table to get the fit and fill weights used during the run
271 38 Application.SetOption "Confirm Action Queries", False
272 39 db1.Execute "CREATE TABLE CONSTANTS (ConstantName TEXT, intValue
273 40 DoCmd.TransferText acImportFixed, "Perm_Data Export Specification",
274 41 CONSTANTS, "c:\tblForm\Temp\Constant.tbl")
275 41 Application.SetOption "Confirm Action Queries", True
276 42
277 43 strSQL = "SELECT intValue FROM CONSTANTS;"
278 44 Set rec = db1.OpenRecordset(strSQL, dbOpenSnapshot)
279 45 Me.tblFWWeight = rec.intValue
280 46 rec.MoveNext
281 47 rec.MoveNext
282 48 Me.tblFWWeight = rec.intValue
283 49 rec.Close
284 50
285 51 ' Remove the created table
286 52 db1.Execute "DROP TABLE CONSTANTS;"
287 53 db1.Close
288 54
289 55
290 56 End Sub
291 57 Private Sub Close_Click()
292 58 On Error GoTo Err_Close_Click
293 59
294 60 Dim stDocName As String
295 61 Dim stLinkCriteria As String
296 62
297 63 ' Close the current form
298 64 DoCmd.Close
299 65 ' Open specified form
300 66 stDocName = "frmAnalysisResult"
301 67 DoCmd.OpenForm stDocName, , stLinkCriteria
302 68
303 69 Exit_Close_Click
304 70 Exit Sub
305 71
306 72 Err_Close_Click
307 73 MsgBox Err.Description
308 74 Resume Exit_Close_Click
309 75
310 76 End Sub
311
312 Form: frmChangeGradDate
313 Code
314 1 Attribute VB_Name = "Form_frmChangeGradDate"
315 2 Attribute VB_Creatable = True
316 3 Attribute VB_PredeclaredId = True
317 4 Attribute VB_Exposed = False
318 5 Option Compare Database
319 6 Option Explicit
320 7
321 8 Private Sub btnChangeDate_Click()
322 9
323 10 ' Change the specified graduation date in the MARINE table
324 11 Application.SetOption "Confirm Action Queries", False
325 12 DoCmd.OpenQuery "qryChangeGradDate"
326 13 Application.SetOption "Confirm Action Queries", True
327 14
328 15 ' Update the drop down list for MCRD Grad Date
329 16 Me.cmbMCRDGradDate.RowSource = "SELECT DISTINCTROW
330 17 [qryMCTGradDate].[GradDate] FROM [qryMCTGradDate];"
331 17
332 18 ' Force user to refresh data before performing another date change
333 19 Me.tblMCTGradDate.Enabled = False
334 20 Me.tblMCTGradDate.Enabled = False
335 21 Me.tblMCTGradDate.Enabled = False
336 22
337 23 ' Provide a message indicating the change was made
338 24 MsgBox "Date change was successful!"
339 25
340 26 End Sub
341 27
342 28 Private Sub btnEquals_Click()
343 29
344 30 Dim varDate As Variant
345 31 Dim inDays As Integer
346 32
347 33 inDays = Me.cmbDays
348 34 varDate = Me.cmbMCRDGradDate
349 35
350 36 If IsNull(varDate) = False And IsNull(inDays) = False Then
351 37
352 38 ' Format this date for use in the DateAdd function
353 39 varDate = Format(varDate, "####/##/##")
354 40
355 41 ' Add 120 days to this date
356 42 varDate = DateAdd("d", inDays, varDate)
357 43
358 44 ' Change calendar to this date
359 45 Me.tblCalendar.Value = varDate
360 46
361 47 ' Convert the new date back into the original format
362 48 varDate = Format(varDate, "yyyymmdd")
363 49
364 50 ' Place the new date in the "upper date bound" text box on the
365 calling form.
366 51 Me.tblMCTGradDate = varDate
367 52
368 53 ' Allows user to change the graduation date in the MARINE table
369 54 Me.tblMCTGradDate.Enabled = True
370 55 End If
371 56
372 57 End Sub
373 58
374 59 Private Sub cmbMCRDGradDate_AfterUpdate()
375 60
376 61 Dim varDate As Variant
377 62
378 63 varDate = Me.cmbMCRDGradDate
379 64 Me.tblMCTGradDate.Value = ""
380 65
381 66 If IsNull(varDate) = False Then
382 67
383 68 ' Format this date for use in the DateAdd function
384 69 varDate = Format(varDate, "####/##/##")
385 70
386 71 ' Change calendar to this date
387 72 Me.tblCalendar.Value = varDate
388 73 Me.tblCalendar.Visible = True
389 74 Me.tblMCTGradDate.Enabled = True
390 75 Me.tblMCTGradDate.Enabled = True

```

```

391 76 Me.tblMCTGradDate.Enabled = True
392 77
393 78 End If
394 79
395 80 End Sub
396 81
397 82 Private Sub Form_Current()
398 83
399 84 Me.tblCalendar.Visible = False
400 85 Me.tblMCTGradDate.Enabled = False
401 86 Me.tblMCTGradDate.Enabled = False
402 87 Me.tblMCTGradDate.Enabled = False
403 88 Me.tblMCTGradDate.Enabled = False
404 89
405 90 End Sub
406 91 Private Sub Close_Click()
407 92 On Error GoTo Err_Close_Click
408 93
409 94 Dim stDocName As String
410 95 Dim stLinkCriteria As String
411 96
412 97 ' Close the current form
413 98 DoCmd.Close
414 99 ' Open specified form
415 100 stDocName = "frmImportExportSwitchboard"
416 101 DoCmd.OpenForm stDocName, , stLinkCriteria
417 102
418 103 Exit_Close_Click
419 104 Exit Sub
420 105
421 106 Err_Close_Click
422 107 MsgBox Err.Description
423 108 Resume Exit_Close_Click
424 109
425 110 End Sub
426
427 Form: frmClass
428 Code
429 1 Attribute VB_Name = "Form_frmClass"
430 2 Attribute VB_Creatable = True
431 3 Attribute VB_PredeclaredId = True
432 4 Attribute VB_Exposed = False
433 5 Option Compare Database
434 6 Option Explicit
435 7
436 8 Private Sub btnFindSchool_Click()
437 9 On Error GoTo Err_btnFindSchool_Click
438 10
439 11
440 12 Screen.PreviousControl.SetFocus
441 13 DoCmd.DoMenuItem acFormBar, acEditMenu, 10, , acMenuVer70
442 14
443 15 Exit_btnFindSchool_Click
444 16 Exit Sub
445 17
446 18 Err_btnFindSchool_Click
447 19 MsgBox Err.Description
448 20 Resume Exit_btnFindSchool_Click
449 21
450 22 End Sub
451
452 Form: frmClassQuotaPenaltyAndFit
453 Code
454 1 Attribute VB_Name = "Form_frmClassQuotaPenaltyAndFit"
455 2 Attribute VB_Creatable = True
456 3 Attribute VB_PredeclaredId = True
457 4 Attribute VB_Exposed = False
458 5 Option Compare Database
459 6 Option Explicit
460 7
461 8 Private Sub btnGo_Click()
462 9
463 10 Dim varUBound
464 11 Dim lngHour As Long, lngMin As Long, lngSec As Long
465 12 Dim lngStart As Long, lngEnd As Long
466 13 Dim strSQL As String
467 14
468 15 ' Starts the timer for displaying the time taken for this subroutine
469 16 lngStart = Timer
470 17
471 18
472 19 ' Prevents the user from going back to change the MCTGradDate until a
473 fitness file is
474 20 ' made, or the reset button is pushed.
475 21 Me.tblMCTGradDate.Enabled = False
476 22 varUBound = Me.UBound
477 23
478 24 ' Generate the quota and penalty files
479 25 Quota_Penalty (varUBound)
480 26
481 27 ' Update the quota/penalty records on this form
482 28 strSQL = "SELECT DISTINCTROW [CLASS].[SCourseNumber_FK] &
483 29 '[CLASS].[ANODE_FK].[CLASS].[ClassNumber_FK].[CLASS].[ReportDate],
484 30 '[CLASS].[Quota], [CLASS].[ClassIndex], [PENALTY].[ClassIndex_FK],
485 31 '[PENALTY].[Penalty] FROM [CLASS] INNER JOIN [PENALTY] ON " &
486 32 "[CLASS].[ClassIndex] = [PENALTY].[ClassIndex_FK];"
487 33 Me.RecordSource = strSQL
488 34
489 35 ' update status of status text box
490 36 Me.Status = "Working..."
491 37
492 38 ' Call the Fitness procedure
493 39 Fitness
494 40
495 41 ' update status of status text box
496 42 Me.Status = "Files generated"
497 43 DoCmd.Beep
498 44
499 45 ' Records the stop time for running this subroutine
500 46 lngEnd = Timer
501 47 ' Determine total secs
502 48 lngSec = lngEnd - lngStart
503 49 ' Determine total hours
504 50 lngHour = lngSec \ 3600
505 51 ' Determine total mins
506 52 lngMin = (lngSec - lngHour * 3600) \ 60
507 53 ' Determine the secs remaining
508 54 lngSec = (lngSec - (lngMin * 60) - (lngHour * 3600)) \ 1
509 55 MsgBox "This operation took " & lngHour & " hour(s), " & lngMin &
510 56 " and " & lngSec & " secs.", vbOKOnly, "Recruit Distribution Model"
511 57
512 58 Me.tblMCTGradDate.Enabled = True
513 59 Me.tblMCTGradDate.Enabled = True
514 60
515 61 End Sub
516 62
517 63 Private Sub btnGo_MouseDown(Button As Integer, Shift As Integer, X As
518 64 Y As Single)
519 65
520 66

```

```

521 65 Me!Status = "Working..."
522 67
523 68 End Sub
524 69
525 70 Private Sub btnPrepareAndExecuteSolve_Click()
526 71 On Error GoTo Err_btnPrepareAndExecuteSolve_Click
527 72
528 73 Dim stDocName As String
529 74 Dim stLinkCriteria As String
530 75
531 76 ' Close the current form
532 77 DoCmd.Close
533 78 ' Open specified form
534 79 stDocName = "frmPrepareAndExecuteSolve"
535 80 DoCmd.OpenForm stDocName, , stLinkCriteria
536 81
537 82 Exit_btnPrepareAndExecuteSolve_Click:
538 83 Exit Sub
539 84
540 85 Err_btnPrepareAndExecuteSolve_Click:
541 86 MsgBox Err.Description
542 87 Resume Exit_btnPrepareAndExecuteSolve_Click
543 88
544 89 End Sub
545 90
546 91 Private Sub btnReturnPrevious_Click()
547 92 On Error GoTo Err_btnReturnPrevious_Click
548 93
549 94 Dim stDocName As String
550 95 Dim stLinkCriteria As String
551 96
552 97 ' Close the current form
553 98 DoCmd.Close
554 99 ' Open specified form
555 100 stDocName = "frmPreprocessingAndExecutionSwitchboard"
556 101 DoCmd.OpenForm stDocName, , stLinkCriteria
557 102
558 103 Exit_btnReturnPrevious_Click:
559 104 Exit Sub
560 105
561 106 Err_btnReturnPrevious_Click:
562 107 MsgBox Err.Description
563 108 Resume Exit_btnReturnPrevious_Click
564 109
565 110 End Sub
566 111
567 112 Private Sub cmbCourseNumberFind_AfterUpdate()
568 113
569 114 Dim R As Recordset
570 115 Set R = Me.RecordsetClone
571 116 R.FindFirst "[SCourseNumber_PK] = " & Chr(34) &
572 117 Me.Bookmark & Chr(34)
573 118 Me!cmbCourseNumberFind = Null
574 119 Me.Penalty.SetFocus
575 120
576 121 End Sub
577 122
578 123 Private Sub Form_Current()
579 124
580 125 ' Prevents the user from creating the AMPL files until a MCT
581 126 ' graduation date is entered
582 127 If IsNull(Me.MCTGradDate.Value) Then
583 128 Me!btnGo.Enabled = False
584 129 Me!Status.Enabled = False
585 130
586 131 ' Allows the user to create the quote and penalty files
587 132 Else
588 133 Me!btnGo.Enabled = True
589 134
590 135 End If
591 136 ' Update the status of the status text box
592 137 Me!Status = ""
593 138
594 139 End Sub
595 140
596 141 Private Sub MCTGradDate_AfterUpdate()
597 142
598 143 Dim varDate As Variant
599 144
600 145 varDate = Me.MCTGradDate
601 146
602 147 If IsNull(varDate) = False Then
603 148
604 149 Me!btnGo.Enabled = True
605 150 Me!Status.Enabled = True
606 151
607 152 ' Format this date for use in the DateAdd function
608 153 varDate = Format(varDate, "###/##/##")
609 154
610 155 ' Add 120 days to this date
611 156 varDate = DateAdd("d", 120, varDate)
612 157
613 158 ' Convert the new date back into the original format
614 159 varDate = Format(varDate, "yyyymmdd")
615 160
616 161 ' Place the new date in the "upper date bound" text box on the
617 162 ' calling form.
618 163 Me.UDateBound = varDate
619 164
620 165 Else
621 166 Me!btnGo.Enabled = False
622 167 Me!Status.Enabled = False
623 168 End If
624 169
625 170 End Sub
626 171
627 172
628 173
629 Form: frmFind
630 Code
631 1 Attribute VB_Name = "Form_frmFind"
632 2 Attribute VB_Creatable = True
633 3 Attribute VB_PredeclaredId = True
634 4 Attribute VB_Exposed = False
635 5 Option Compare Database
636 6 Option Explicit
637 7
638 8 Private Sub btnFind_Click()
639 9 On Error GoTo Err_btnFind_Click
640 10
641 11 'Screen.PreviousControl.SetFocus
642 12 Form!frmSchoolToPropertyMaint.SetFocus
643 13 DoCmd.FindRecord Me.cmbCourseNumber, , True, , True, acAll
644 14 DoCmd.DoMenuItem acFormBar, acEditMenu, 10, , acMenuVer70
645 15
646 16 Exit_btnFind_Click:
647 17 Exit Sub
648 18
649 19 Err_btnFind_Click:
650 20 MsgBox Err.Description

```

```

651 21 Resume Exit_btnFind_Click
652 22 End Sub
653 23
654 24
655 25
656 Form: frmFundamentalProperty
657 Code
658 1 Attribute VB_Name = "Form_frmFundamentalProperty"
659 2 Attribute VB_Creatable = True
660 3 Attribute VB_PredeclaredId = True
661 4 Attribute VB_Exposed = False
662 5 Option Compare Database
663 6 Option Explicit
664 7
665 8 Private Sub Command15_Click()
666 9
667 10 MsgBox "This is a button, no information, 'Testing'"
668 11
669 12 End Sub
670 13
671 14 Private Sub btnClose_Click()
672 15 On Error GoTo Err_btnClose_Click
673 16
674 17 DoCmd.Close
675 18 DoCmd.OpenForm "frmMaintenanceSwitchboard"
676 19
677 20 Exit_btnClose_Click:
678 21 Exit Sub
679 22
680 23 Err_btnClose_Click:
681 24 MsgBox Err.Description
682 25 Resume Exit_btnClose_Click
683 26
684 27 End Sub
685 28
686 29
687 30
688 31 Private Sub cmbPropertyNameFind_AfterUpdate()
689 32
690 33 Dim R As Recordset
691 34 Set R = Me.RecordsetClone
692 35 R.FindFirst "[FPropertyName_PK] = " & Chr(34) &
693 36 Me!cmbPropertyNameFind & Chr(34)
694 37 Me!Bookmark = R.Bookmark
695 38 Me!cmbPropertyNameFind = Null
696 39 Me.FPropertyName_PK.SetFocus
697 40
698 41 End Sub
699 42
700 43 Private Sub Command11_Click()
701 44 ' Forces user to enter data for property name and description
702 45 If IsNull(Me.FPropertyName_PK.Value) Or IsNull(Me.Description.Value)
703 46 Me!MarineField.Enabled = False
704 47 Me!Operator.Enabled = False
705 48 Me!Label10.Visible = False
706 49 Me!subfrmFundamentalProperty.Visible = False
707 50 Me!subfrmFundamentalPropertyList.Visible = False
708 51 Me!cmbPotentialValue.Visible = False
709 52 Me!Label10.Visible = False
710 53 ' Allows the user to enter values for the fundamental equation
711 54 Else
712 55 Me!MarineField.Enabled = True
713 56 Me!Operator.Enabled = True
714 57 Me!Label10.Visible = True
715 58 If Format!frmFundamentalProperty.Operator.Value = "in" Then
716 59 Me!subfrmFundamentalProperty.Visible = False
717 60 Me!subfrmFundamentalPropertyList.Visible = True
718 61
719 62 Else
720 63 Me!cmbPotentialValue.Visible = True
721 64 Me!subfrmFundamentalProperty.Visible = True
722 65 Me!subfrmFundamentalPropertyList.Visible = False
723 66 End If
724 67 End If
725 68
726 69 End Sub
727 70
728 71 Private Sub Description_AfterUpdate()
729 72
730 73 Dim stSQL As String
731 74
732 75 ' Forces user to enter data for property name and description
733 76 If IsNull(Me.FPropertyName_PK.Value) Or IsNull(Me.Description.Value)
734 77 Me!MarineField.Enabled = False
735 78 Me!Operator.Enabled = False
736 79 Me!Label10.Visible = False
737 80 Me!subfrmFundamentalProperty.Visible = False
738 81 Me!subfrmFundamentalPropertyList.Visible = True
739 82
740 83 ' Allows the user to enter values for the fundamental equation
741 84 Else
742 85 Me!MarineField.Enabled = True
743 86 Me!Operator.Enabled = True
744 87 Me!Label10.Visible = True
745 88 If Format!frmFundamentalProperty.Operator.Value = "in" Then
746 89 stSQL = "SELECT DISTINCT " & Me!MarineField & " FROM
747 90 Form!frmFundamentalProperty!subfrmFundamentalPropertyList!Value_FK.RowSo
748 91 urce = stSQL
749 92 Me!subfrmFundamentalProperty.Visible = False
750 93 Me!subfrmFundamentalPropertyList.Visible = True
751 94
752 95 Else
753 96 stSQL = "SELECT DISTINCT " & Me!MarineField & " FROM
754 97 Form!frmFundamentalProperty!subfrmFundamentalProperty!cmbValue_FK.RowSou
755 98 rce = stSQL
756 99 Me!subfrmFundamentalProperty.Visible = True
757 100 Me!subfrmFundamentalPropertyList.Visible = False
758 101 End If
759 102
760 103 End Sub
761 104 Private Sub Form_BeforeUpdate(Cancel As Integer)
762 105
763 106 [FundPropTimeStamp] = Now
764 107
765 108 End Sub
766 109
767 110 Private Sub Form_Current()
768 111
769 112 Dim stSQL As String
770 113
771 114 stSQL = "SELECT MarineField, Operator FROM FUNDAMENTAL_PROPERTY
772 115 WHERE FPropertyName_PK = " & FundPropName
773 116
774 117
775 118
776 119
777 120
778 121
779 122
780 123

```



```

781 117 ' Forces user to enter data for property name and description
782 118 If IsNull(Me.FPropertyName_PK.Value) Or IsNull(Me.Description.Value)
783 119 Me.MarineField.Enabled = False
784 120 Me.Operator.Enabled = False
785 121 Me.Labot10.Visible = False
786 122 Me.SubfrmFundamentaProperty.Visible = False
787 123 Me.SubfrmFundamentaPropertyList.Visible = False
788 124
789 125 ' Allows the user to enter values for the fundamental equation
790 126 Else
791 127 Me.MarineField.Enabled = True
792 128 Me.Operator.Enabled = True
793 129 Me.Labot10.Visible = True
794 130 If Form!frmFundamentaProperty.Operator.Value = "in" Then
795 131 ' Update the Potential Value list
796 132 Me.SubfrmFundamentaProperty.Visible = False
797 133 Me.SubfrmFundamentaPropertyList.Visible = True
798 134 strSQL = "SELECT DISTINCT * & Me.MarineField & " FROM
799 135
800 Form!frmFundamentaProperty.SubfrmFundamentaPropertyList.Value_FK.RowSou
801 urce = strSQL
802 136 Else
803 137 ' Update the Potential Value list
804 138 strSQL = "SELECT DISTINCT * & Me.MarineField & " FROM
805 139
806 Form!frmFundamentaProperty.SubfrmFundamentaProperty.cmbValue_FK.RowSou
807 rce = strSQL
808 140 Me.SubfrmFundamentaProperty.Visible = True
809 141 Me.SubfrmFundamentaPropertyList.Visible = False
810 142 End If
811 143
812 144 ' Set focus to the property name. This prevents problem
813 associated with entering the "Operator" object.
814 145 FPropertyName_PK.SetFocus
815 146 End If
816 147
817 148
818 149
819 150 End Sub
820 151
821 152 Private Sub FPropertyName_PK_AfterUpdate()
822 153
823 154 Dim strSQL As String
824 155
825 156 ' Forces user to enter data for property name and description
826 157 If IsNull(Me.FPropertyName_PK.Value) Or IsNull(Me.Description.Value)
827 158 Me.MarineField.Enabled = False
828 159 Me.Operator.Enabled = False
829 160 Me.Labot10.Visible = False
830 161 Me.SubfrmFundamentaProperty.Visible = False
831 162 Me.SubfrmFundamentaPropertyList.Visible = False
832 163
833 164 ' Allows the user to enter values for the fundamental equation
834 165 Else
835 166 Me.MarineField.Enabled = True
836 167 Me.Operator.Enabled = True
837 168 Me.Labot10.Visible = True
838 169 If Form!frmFundamentaProperty.Operator.Value = "in" Then
839 170 strSQL = "SELECT DISTINCT * & Me.MarineField & " FROM
840 171
841 Form!frmFundamentaProperty.SubfrmFundamentaPropertyList.Value_FK.RowSou
842 urce = strSQL
843 172 Me.SubfrmFundamentaProperty.Visible = False
844 173 Me.SubfrmFundamentaPropertyList.Visible = True
845 174
846 175 Else
847 176 strSQL = "SELECT DISTINCT * & Me.MarineField & " FROM
848 177
849 Form!frmFundamentaProperty.SubfrmFundamentaProperty.cmbValue_FK.RowSou
850 rce = strSQL
851 178 Me.SubfrmFundamentaProperty.Visible = True
852 179 Me.SubfrmFundamentaPropertyList.Visible = False
853 180 End If
854 181 End If
855 182
856 183 End Sub
857 184
858 185 Private Sub MarineField_AfterUpdate()
859 186
860 187 Dim strSQL As String
861 188
862 189 ' Update the Potential Value list
863 190 strSQL = "SELECT DISTINCT * & Me.MarineField & " FROM MARINE;"
864 191
865 Form!frmFundamentaProperty.SubfrmFundamentaPropertyList.Value_FK.RowSou
866 urce = strSQL
867 192
868 Form!frmFundamentaProperty.SubfrmFundamentaProperty.cmbValue_FK.RowSou
869 rce = strSQL
870 193
871 194 End Sub
872 195
873 196 Private Sub Operator_AfterUpdate()
874 197
875 198 If Form!frmFundamentaProperty.Operator.Value = "in" Then
876 199 Me.SubfrmFundamentaProperty.Visible = False
877 200 Me.SubfrmFundamentaPropertyList.Visible = True
878 201 Else
879 202 Me.SubfrmFundamentaProperty.Visible = True
880 203 Me.SubfrmFundamentaPropertyList.Visible = False
881 204 End If
882 205
883 206 End Sub
884 207 Private Sub btnDelete_Click()
885 208 On Error GoTo Err_btnDelete_Click
886 209
887 210
888 211 DoCmd.DoMenuItem acFormBar, acEditMenu, 8, , acMenuVer70
889 212 DoCmd.DoMenuItem acFormBar, acEditMenu, 6, , acMenuVer70
890 213
891 214 Exit btnDelete_Click
892 215 Exit Sub
893 216
894 217 Err_btnDelete_Click
895 218 MsgBox Err.Description
896 219 Resume Exit_btnDelete_Click
897 220
898 221 End Sub
899 222 Private Sub Command14_Click()
900 223 On Error GoTo Err_Command14_Click
901 224
902 225 Dim stDiaStr As String
903 226 Dim PrvCt As Control
904 227 Const ERR_OBNOTEXIST = 2467
905 228 Const ERR_OBNOTSET = 91
906 229
907 230 Set PrvCt = Screen.PreviousControl
908 231
909 232 If TypeOf PrvCt Is TextBox Then
910 233 stDiaStr = If((VarType(PrvCt) > V_NULL, PrvCt, ""))
911 234 ElseIf TypeOf PrvCt Is ListBox Then
912 235 stDiaStr = If((VarType(PrvCt) > V_NULL, PrvCt, ""))
913 236 ElseIf TypeOf PrvCt Is ComboBox Then
914 237 stDiaStr = If((VarType(PrvCt) > V_NULL, PrvCt, ""))
915 238 Else
916 239 stDiaStr = ""
917 240 End If
918 241
919 242 Application.Run "Utility.web_AutoDia", stDiaStr
920 243
921 244 Exit Command14_Click
922 245 Exit Sub
923 246
924 247 Err_Command14_Click
925 248 If (Err = ERR_OBNOTEXIST) Or (Err = ERR_OBNOTSET) Then
926 249 Resume Next
927 250 End If
928 251 MsgBox Err.Description
929 252 Resume Exit_Command14_Click
930 253
931 254 End Sub
932 255
933 256 Private Sub Operator_Enter()
934 257
935 258 If Me.Operator.Value = "in" Then
936 259 MsgBox "The 'in' operator is not changeable. It is removed by"
937 260 deleting the property. , vbInformation, "Unauthorized Action"
938 261 subfrmFundamentaPropertyList.SetFocus
939 262 End If
940 263 End Sub
941 264
942 265
943 Form: frmGenerateReport
944 Code
945 1 Attribute VB_Name = "Form_frmGenerateReport"
946 2 Attribute VB_Creatable = True
947 3 Attribute VB_PredeclaredId = True
948 4 Attribute VB_Exposed = False
949 5 Option Compare Database
950 6 Option Explicit
951 7
952 8 Private Sub btnAssignmentReport_Click()
953 9
954 10 DoCmd.OpenReport "rptAssignedMarines", acViewPreview
955 11
956 12 End Sub
957 13
958 14 Private Sub btnEntryLevelSchool_Click()
959 15
960 16 DoCmd.OpenReport "rptEntryLevelSchool", acViewPreview
961 17
962 18 End Sub
963 19 End Sub
964 20
965 21 Private Sub btnExit_Click()
966 22 On Error GoTo Err_btnExit_Click
967 23
968 24
969 25 DoCmd.Close
970 26
971 27 Exit btnExit_Click
972 28 Exit Sub
973 29
974 30 Err_btnExit_Click
975 31 MsgBox Err.Description
976 32 Resume Exit_btnExit_Click
977 33
978 34
979 35 End Sub
980 36
981 37 Private Sub btnUnassignedMarineReport_Click()
982 38
983 39 DoCmd.OpenReport "rptUnassignedMarines", acViewPreview
984 40
985 41 End Sub
986 42
987 43 Private Sub Command8_Click()
988 44 On Error GoTo Err_Command8_Click
989 45
990 46 Dim stDocName As String
991 47 Dim stLinkCriteria As String
992 48
993 49 ' Close the current form
994 50 DoCmd.Close
995 51 ' Open specified form
996 52 stDocName = "frmFORM_Main_Switchboard"
997 53 DoCmd.OpenForm stDocName, , stLinkCriteria
998 54
999 55 Exit Command8_Click
1000 56 Exit Sub
1001 57
1002 58 Err_Command8_Click
1003 59 MsgBox Err.Description
1004 60 Resume Exit_Command8_Click
1005 61
1006 62 End Sub
1007
1008 Form: frmImport/ExportSwitchboard
1009 Code
1010 1 Attribute VB_Name = "Form_frmImport/ExportSwitchboard"
1011 2 Attribute VB_Creatable = True
1012 3 Attribute VB_PredeclaredId = True
1013 4 Attribute VB_Exposed = False
1014 5 Option Compare Database
1015 6 Option Explicit
1016 7
1017 8 Private Sub btnExit_Click()
1018 9 On Error GoTo Err_btnExit_Click
1019 10
1020 11
1021 12 DoCmd.Close
1022 13
1023 14 Exit btnExit_Click
1024 15 Exit Sub
1025 16
1026 17 Err_btnExit_Click
1027 18 MsgBox Err.Description
1028 19 Resume Exit_btnExit_Click
1029 20
1030 21
1031 22 End Sub
1032 23
1033 24 Private Sub btnExportR03_Click()
1034 25
1035 26 Application.SetOption "Confirm Action Queries", False
1036 27 DoCmd.SendWarnings (False)
1037 28 ' Exports assignments to R03.txt file
1038 29 ExportR03
1039 30 ' Archives data from the Marine and assignment tables, having report
1040 31 dates less than the current date

```

```

1041 31 ArchiveData
1042 32 Application.SetOption "Confirm Action Queries", True
1043 33 DoCmd.SetWarnings (True)
1044 34
1045 35 End Sub
1046 36
1047 37
1048 38 Private Sub Command2_Click()
1049 39
1050 40 Dim stDocName As String
1051 41 Dim stLinkCriteria As String
1052 42
1053 43 Application.SetOption "Confirm Action Queries", False
1054 44 DoCmd.SetWarnings (False)
1055 45 'Imports the RDT file
1056 46 ImportRDT
1057 47 Application.SetOption "Confirm Action Queries", True
1058 48 DoCmd.SetWarnings (True)
1059 49
1060 50 'Close the current form
1061 51 DoCmd.Close
1062 52 'Open specified form
1063 53 stDocName = "frmChangeGradDate"
1064 54 DoCmd.OpenForm stDocName, , , stLinkCriteria
1065 55
1066 56 End Sub
1067 57
1068 58 Private Sub Command4_Click()
1069 59
1070 60 'Imports BNA.tbl file
1071 61 Application.SetOption "Confirm Action Queries", False
1072 62 DoCmd.SetWarnings (False)
1073 63 DoCmd.RunSQL "DELETE FROM BNA_EXTRACT;"
1074 64 ImportBNA
1075 65 Application.SetOption "Confirm Action Queries", True
1076 66 DoCmd.SetWarnings (True)
1077 67
1078 68 End Sub
1079 69
1080 70 Private Sub Command5_Click()
1081 71
1082 72 On Error GoTo Err_Command5_Click
1083 73
1084 74 Dim stDocName As String
1085 75 Dim stLinkCriteria As String
1086 76
1087 77 'Close the current form
1088 78 DoCmd.Close
1089 79 'Open specified form
1090 80 stDocName = "frmRDM_Main_Switchboard"
1091 81 DoCmd.OpenForm stDocName, , , stLinkCriteria
1092 82
1093 83 Exit_Command5_Click
1094 84 Exit Sub
1095 85
1096 86 Err_Command5_Click:
1097 87 MsgBox Err.Description
1098 88 Resume Exit_Command5_Click
1099 89
1100 90 End Sub
1101
1102 Form: frmLogicalProperty
1103 Code
1104 1 Attribute VB_Name = "Form_frmLogicalProperty"
1105 2 Attribute VB_Creatable = True
1106 3 Attribute VB_PredeclaredId = True
1107 4 Attribute VB_Exposed = False
1108 5 Option Compare Database
1109 6 Option Explicit
1110 7
1111 8 Private Sub btnClose_Click()
1112 9 On Error GoTo Err_btnClose_Click
1113 10
1114 11
1115 12 DoCmd.Close
1116 13 DoCmd.OpenForm "frmMaintenanceSwitchboard"
1117 14
1118 15 Exit_btnClose_Click
1119 16 Exit Sub
1120 17
1121 18 Err_btnClose_Click:
1122 19 MsgBox Err.Description
1123 20 Resume Exit_btnClose_Click
1124 21
1125 22 End Sub
1126 23
1127 24 Private Sub cmbFundProp_Click()
1128 25
1129 26 'Adds the selected fundamental property to the logical equation
1130 27 Me.LogicalEquation.Value = Me.LogicalEquation.Value &
1131 28 Me.cmbFundProp.Value & " "
1132 29
1133 30 End Sub
1134 31 Private Sub cmbLogOper_Click()
1135 32
1136 33 'Adds the selected logical operator to the logical equation
1137 34 Me.LogicalEquation.Value = Me.LogicalEquation.Value &
1138 35 Me.cmbLogOper.Value & " "
1139 36
1140 37 End Sub
1141 38
1142 39 Private Sub cmbPropertyNameFind_AfterUpdate()
1143 40
1144 41 Dim R As Recordset
1145 42 Set R = Me.RecordsetClone
1146 43 R.FindFirst "[!PropertyName_PK] = " & Chr(34) &
1147 44 Me.cmbPropertyNameFind & Chr(34)
1148 45 Me.Bookmark = R.Bookmark
1149 46 Me!(cmbPropertyNameFind) = Null
1150 47 Me.LPPropertyName_PK.SetFocus
1151 48
1152 49 End Sub
1153 50
1154 51 Private Sub Command10_Click()
1155 52
1156 53 'Removes the last entry in the logical equation
1157 54 Dim stStrName, strUndo As String
1158 55 Dim intSpacePos As Integer
1159 56 Dim i As Integer
1160 57 Dim boolResult As Boolean
1161 58 Dim lngConversion As Long
1162 59 Dim varConv As Variant
1163 60
1164 61 'get the logical expression
1165 62 strName = Me.LogicalEquation.Value
1166 63 'check for "Null" value
1167 64 If strName <> "Null" Then
1171 65
1172 66 'declare the temporary string as an array
1173 67 ReDim strTemp(0) As String
1174 68
1175 69 'Find the position of the first space
1176 70 intSpacePos = InStr(strName, " ")
1177 71
1178 72 'Loop until no more spaces are found. InStr returns 0 if the
1179 73 space is not found.
1180 74 Do Until intSpacePos = 0
1181 75 'redimension the array, adding another element
1182 76 'UBound tells us how many elements there already are in the
1183 77 ReDim Preserve strTemp(UBound(strTemp) + 1)
1184 78
1185 79 'set the new element to the characters from the beginning of
1186 80 'string up to the first space. Left returns a number of
1187 81 characters
1188 82 strTemp(UBound(strTemp)) = Trim(Left(strName, intSpacePos))
1189 83
1190 84 'now we've copied these characters into the name, they can
1191 85 'be removed from the original string. Right returns a number
1192 86 'characters from the end of the string. It is the rightmost
1193 87 'characters, therefore we subtract the position of the space
1194 88 'the length of the string. This gives us the number of
1195 89 'characters remaining
1196 90 strName = Trim(Right(strName, Len(strName) -
1197 91 intSpacePos))
1198 92
1199 93 'find the next space
1200 94 intSpacePos = InStr(strName, " ") Or InSpacePos =
1201 95 InStr(strName, "(") Or InSpacePos = InStr(strName, ")")
1202 96
1203 97 Loop
1204 98
1205 99 'Remove the last element of the array, unless there is only one
1206 100 element left
1207 101 If UBound(strTemp) > 1 Then
1208 102 ReDim Preserve strTemp(UBound(strTemp) - 1)
1209 103
1210 104 'Loop through the elements of the array and place them back
1211 105 into the logical equation expression
1212 106 For i = 1 To (UBound(strTemp))
1213 107 strUndo = strUndo & strTemp(i) & " "
1214 108 Me.LogicalEquation.Value = strUndo
1215 109 Next i
1216 110 Else
1217 111 'Set the logical equation equal to "Null"
1218 112 Me.LogicalEquation.Value = "Null"
1219 113 End If
1220 114
1221 115 'Set logical equation expression equal to new value
1222 116 End If
1223 117
1224 118 End Sub
1225 119
1226 120 Private Sub Command11_Click()
1227 121
1228 122 'Forces user to enter data for property name and description
1229 123 If IsNull(Me.LPPropertyName_PK.Value) Or IsNull(Me.Description.Value)
1230 124 Me.Command8.Enabled = False
1231 125 Me.Command9.Enabled = False
1232 126 Me.Command10.Enabled = False
1233 127 Me.cmbFundProp.Enabled = False
1234 128 Me.cmbLogOper.Enabled = False
1235 129 Me.LogicalEquation.Enabled = False
1236 130 MsgBox "You must provide a 'Property Name' and 'Description'"
1237 131 before building the logical equation.
1238 132
1239 133 'Enters a "(" be the first entry in the logical equation if it
1240 134 isNull(Me.LogicalEquation.Value) Then
1241 135 Me.Command8.Enabled = True
1242 136 Me.Command9.Enabled = True
1243 137 Me.Command10.Enabled = True
1244 138 Me.cmbFundProp.Enabled = True
1245 139 Me.cmbLogOper.Enabled = True
1246 140 Me.LogicalEquation.Enabled = True
1247 141 Me.LogicalEquation.Value = "("
1248 142
1249 143 'Allows the user to enter values into the logical equation
1250 144 Else
1251 145 Me.Command8.Enabled = True
1252 146 Me.Command9.Enabled = True
1253 147 Me.Command10.Enabled = True
1254 148 Me.cmbFundProp.Enabled = True
1255 149 Me.cmbLogOper.Enabled = True
1256 150 Me.LogicalEquation.Enabled = True
1257 151 Me.LogicalEquation.Value = Me.LogicalEquation.Value & "("
1258 152
1259 153 End Sub
1260 154
1261 155 'Adds the left bracket to the logical equation
1262 156 Me.LogicalEquation.Value = Me.LogicalEquation.Value & "("
1263 157
1264 158 End Sub
1265 159
1266 160 Private Sub btnDelete_Click()
1267 161 On Error GoTo Err_btnDelete_Click
1268 162
1269 163 DoCmd.DoMenuItem acFormBar, acEditMenu, 8, , acMenuVer70
1270 164 DoCmd.DoMenuItem acFormBar, acEditMenu, 6, , acMenuVer70
1271 165
1272 166 Exit_btnDelete_Click
1273 167 Exit Sub
1274 168
1275 169 Err_btnDelete_Click:
1276 170 MsgBox Err.Description
1277 171 Resume Exit_btnDelete_Click
1278 172
1279 173 End Sub
1280 174
1281 175 Private Sub Description_AfterUpdate()
1282 176
1283 177 'Forces user to enter data for property name and description
1284 178 If IsNull(Me.LPPropertyName_PK.Value) Or IsNull(Me.Description.Value)
1285 179 Me.Command8.Enabled = False
1286 180 Me.Command9.Enabled = False
1287 181 Me.Command10.Enabled = False
1288 182 Me.cmbFundProp.Enabled = False
1289 183 Me.cmbLogOper.Enabled = False
1290 184 Me.LogicalEquation.Enabled = False
1291 185
1292 186 'Enters a "(" be the first entry in the logical equation if it
1293 187 isNull(Me.LogicalEquation.Value) Then

```

```

1301 188 Me!Command8.Enabled = True
1302 189 Me!Command9.Enabled = True
1303 190 Me!Command10.Enabled = True
1304 191 Me!cmbFundProp.Enabled = True
1305 192 Me!cmbLogOper.Enabled = True
1306 193 Me!LogicalEquation.Enabled = True
1307 194 Me!LogicalEquation.Value = "("
1308 195
1309 196 ' Allows the user to enter values into the logical equation
1310 197 Else
1311 198 Me!Command8.Enabled = True
1312 199 Me!Command9.Enabled = True
1313 200 Me!Command10.Enabled = True
1314 201 Me!cmbFundProp.Enabled = True
1315 202 Me!cmbLogOper.Enabled = True
1316 203 Me!LogicalEquation.Enabled = True
1317 204 End If
1318 205
1319 206 End Sub
1320 207
1321 208 Private Sub Form_BeforeUpdate(Cancel As Integer)
1322 209
1323 210 [LogPropTimeStamp] = Now
1324 211
1325 212 End Sub
1326 213
1327 214 Private Sub Form_Current()
1328 215
1329 216 ' Forces user to enter data for property name and description
1330 217 If IsNull(Me.LPropertyName_PK.Value) Or IsNull(Me.Description.Value)
1331 218 Me!Command8.Enabled = False
1332 219 Me!Command9.Enabled = False
1333 220 Me!Command10.Enabled = False
1334 221 Me!cmbFundProp.Enabled = False
1335 222 Me!cmbLogOper.Enabled = False
1336 223 Me!LogicalEquation.Enabled = False
1337 224
1338 225 ' Enters a "(" be the first entry in the logical equation if it is
1339 226 ElseIf IsNull(Me!LogicalEquation.Value) Then
1340 227 Me!LogicalEquation.Value = "("
1341 228
1342 229 ' Allows the user to enter values into the logical equation
1343 230 Else
1344 231 Me!Command8.Enabled = True
1345 232 Me!Command9.Enabled = True
1346 233 Me!Command10.Enabled = True
1347 234 Me!cmbFundProp.Enabled = True
1348 235 Me!cmbLogOper.Enabled = True
1349 236 Me!LogicalEquation.Enabled = True
1350 237 End If
1351 238
1352 239 End Sub
1353 240 Private Sub LogicalEquation_Enter()
1354 241
1355 242 ' Prevents the user from modifying the logical equation directly
1356 243 MsgBox "The logical equation is only modifiable by the above logical
1357 244 equation builder"
1358 245 DoCmd.GoToControl "LPropertyName_PK"
1359 246
1360 247 End Sub
1361 248
1362 249 Private Sub LPropertyName_PK_AfterUpdate()
1363 250
1364 251 ' Forces user to enter data for property name and description
1365 252 If IsNull(Me.LPropertyName_PK.Value) Or IsNull(Me.Description.Value)
1366 253 Me!Command8.Enabled = False
1367 254 Me!Command9.Enabled = False
1368 255 Me!Command10.Enabled = False
1369 256 Me!cmbFundProp.Enabled = False
1370 257 Me!cmbLogOper.Enabled = False
1371 258 Me!LogicalEquation.Enabled = False
1372 259
1373 260 ' Enters a "(" be the first entry in the logical equation if it is
1374 261 ElseIf IsNull(Me!LogicalEquation.Value) Then
1375 262 Me!Command8.Enabled = True
1376 263 Me!Command9.Enabled = True
1377 264 Me!Command10.Enabled = True
1378 265 Me!cmbFundProp.Enabled = True
1379 266 Me!cmbLogOper.Enabled = True
1380 267 Me!LogicalEquation.Enabled = True
1381 268 Me!LogicalEquation.Value = "("
1382 269
1383 270 ' Allows the user to enter values into the logical equation
1384 271 Else
1385 272 Me!Command8.Enabled = True
1386 273 Me!Command9.Enabled = True
1387 274 Me!Command10.Enabled = True
1388 275 Me!cmbFundProp.Enabled = True
1389 276 Me!cmbLogOper.Enabled = True
1390 277 Me!LogicalEquation.Enabled = True
1391 278 End If
1392 279
1393 280 End Sub
1394 281
1395 282 Form: frmMaintenanceSwitchboard
1396 283 Code
1397 284 1 Attribute VB_Name = "Form_frmMaintenanceSwitchboard"
1398 285 2 Attribute VB_Creatable = True
1399 286 3 Attribute VB_PredeclaredId = True
1400 287 4 Attribute VB_Exposed = False
1401 288 5 Option Compare Database
1402 289 6 Option Explicit
1403 290 7
1404 291 8 Private Sub btnExit_Click()
1405 292 9 On Error GoTo Err_btnExit_Click
1406 293 10
1407 294 11
1408 295 12 DoCmd.Close
1409 296 13
1410 297 14 Exit_btnExit_Click
1411 298 15 Exit Sub
1412 299 16
1413 300 17 Err_btnExit_Click
1414 301 18 MsgBox Err.Description
1415 302 19 Resume Exit_btnExit_Click
1416 303 20
1417 304 21
1418 305 22 End Sub
1419 306 23
1420 307 24 Private Sub Command0_Click()
1421 308 25 On Error GoTo Err_Command0_Click
1422 309 26
1423 310 27 Dim stDocName As String
1424 311 28 Dim stLinkCriteria As String
1425 312 29
1426 313 30 ' Close the current form
1427 314 31 DoCmd.Close
1428 315 32 ' Open specified form
1429 316 33 stDocName = "frmSchoolToPEF_Maint"
1430 317 34 DoCmd.OpenForm stDocName, , , stLinkCriteria

```

```

1431 35
1432 36 Exit_Command0_Click
1433 37 Exit Sub
1434 38
1435 39 Err_Command0_Click
1436 40 MsgBox Err.Description
1437 41 Resume Exit_Command0_Click
1438 42
1439 43 End Sub
1440 44
1441 45 Private Sub Command12_Click()
1442 46 On Error GoTo Err_Command12_Click
1443 47
1444 48 Dim stDocName As String
1445 49 Dim stLinkCriteria As String
1446 50
1447 51 ' Close the current form
1448 52 DoCmd.Close
1449 53 ' Open specified form
1450 54 stDocName = "frmSchoole"
1451 55 DoCmd.OpenForm stDocName, , , stLinkCriteria
1452 56
1453 57 Exit_Command12_Click
1454 58 Exit Sub
1455 59
1456 60 Err_Command12_Click
1457 61 MsgBox Err.Description
1458 62 Resume Exit_Command12_Click
1459 63
1460 64 End Sub
1461 65
1462 66 Private Sub Command14_Click()
1463 67 On Error GoTo Err_Command14_Click
1464 68
1465 69 Dim stDocName As String
1466 70 Dim stLinkCriteria As String
1467 71
1468 72 ' Close the current form
1469 73 DoCmd.Close
1470 74 ' Open specified form
1471 75 stDocName = "frmPEF"
1472 76 DoCmd.OpenForm stDocName, , , stLinkCriteria
1473 77
1474 78 Exit_Command14_Click
1475 79 Exit Sub
1476 80
1477 81 Err_Command14_Click
1478 82 MsgBox Err.Description
1479 83 Resume Exit_Command14_Click
1480 84
1481 85 End Sub
1482 86
1483 87 Private Sub Command2_Click()
1484 88 On Error GoTo Err_Command2_Click
1485 89
1486 90 Dim stDocName As String
1487 91 Dim stLinkCriteria As String
1488 92
1489 93 ' Close the current form
1490 94 DoCmd.Close
1491 95 ' Open specified form
1492 96 stDocName = "frmFundamentalProperty"
1493 97 DoCmd.OpenForm stDocName, , , stLinkCriteria
1494 98
1495 99 Exit_Command2_Click
1496 100 Exit Sub
1497 101
1498 102 Err_Command2_Click
1499 103 MsgBox Err.Description
1500 104 Resume Exit_Command2_Click
1501 105
1502 106 End Sub
1503 107
1504 108 Private Sub Command4_Click()
1505 109 On Error GoTo Err_Command4_Click
1506 110
1507 111 Dim stDocName As String
1508 112 Dim stLinkCriteria As String
1509 113
1510 114 ' Close the current form
1511 115 DoCmd.Close
1512 116 ' Open specified form
1513 117 stDocName = "frmLogicalProperty"
1514 118 DoCmd.OpenForm stDocName, , , stLinkCriteria
1515 119
1516 120 Exit_Command4_Click
1517 121 Exit Sub
1518 122
1519 123 Err_Command4_Click
1520 124 MsgBox Err.Description
1521 125 Resume Exit_Command4_Click
1522 126
1523 127 End Sub
1524 128
1525 129 Private Sub Command6_Click()
1526 130 On Error GoTo Err_Command6_Click
1527 131
1528 132 Dim stDocName As String
1529 133 Dim stLinkCriteria As String
1530 134
1531 135 ' Close the current form
1532 136 DoCmd.Close
1533 137 ' Open specified form
1534 138 stDocName = "frmSchoolToPEF_Maint"
1535 139 DoCmd.OpenForm stDocName, , , stLinkCriteria
1536 140
1537 141 Exit_Command6_Click
1538 142 Exit Sub
1539 143
1540 144 Err_Command6_Click
1541 145 MsgBox Err.Description
1542 146 Resume Exit_Command6_Click
1543 147
1544 148 End Sub
1545 149
1546 150 Private Sub Command8_Click()
1547 151 On Error GoTo Err_Command8_Click
1548 152
1549 153 Dim stDocName As String
1550 154 Dim stLinkCriteria As String
1551 155
1552 156 ' Close the current form
1553 157 DoCmd.Close
1554 158 ' Open specified form
1555 159 stDocName = "frmFDM_Main_Switchboard"
1556 160 DoCmd.OpenForm stDocName, , , stLinkCriteria
1557 161
1558 162 Exit_Command8_Click
1559 163 Exit Sub
1560 164

```

```

1561 165 Err_Command8_Click
1562 166 MsgBox Err.Description
1563 167 Resume Exit_Command8_Click
1564 168
1565 169 End Sub
1566
1567 Form: frmPEF
1568 Code
1569 1 Attribute VB_Name = "Form_frmPEF"
1570 2 Attribute VB_Creatable = True
1571 3 Attribute VB_PredeclaredId = True
1572 4 Attribute VB_Exposed = False
1573 5 Option Compare Database
1574 6 Option Explicit
1575 7
1576 8 Private Sub cmbPeFind_AfterUpdate()
1577 9
1578 10 Dim R As Recordset
1579 11 Set R = Me.RecordsetClone
1580 12 R.FindFirst "[PEF_PK] = " & Chr(34) & Me[cmbPeFind] & Chr(34)
1581 13 Me.Bookmark = R.Bookmark
1582 14 Me[cmbPeFind] = Null
1583 15 Me.PEF_PK.SetFocus
1584 16
1585 17 End Sub
1586
1587 18 Private Sub Form_BeforeUpdate(Cancel As Integer)
1588 19
1589 20
1590 21 [PEF_TimeStamp] = Now
1591 22
1592 23 End Sub
1593
1594 24 Private Sub Form_Current()
1595 25
1596 26 Dim strSQL As String
1597 27
1598 28 ' Update the PEF List
1599 29 strSQL = "SELECT DISTINCTROW [MARINE].[PEF]FROM MARINE LEFT JOIN PEF
1600 30 ON [MARINE].[PEF] = [PEF].[PEF_PK]WHERE ([PEF].[PEF_PK] IS NULL)ORDER BY
1601 31 Me.PEF_PK.RowSource = strSQL
1602 32
1603 33 End Sub
1604
1605 34 Private Sub PEF_PK_AfterUpdate()
1606 35
1607 36 Dim strSQL As String
1608 37
1609 38 ' Update the PEF List
1610 39 strSQL = "SELECT DISTINCTROW [MARINE].[PEF]FROM MARINE LEFT JOIN PEF
1611 40 ON [MARINE].[PEF] = [PEF].[PEF_PK]WHERE ([PEF].[PEF_PK] IS NULL)ORDER BY
1612 41 Me.PEF_PK.RowSource = strSQL
1613 42
1614 43 End Sub
1615 44 Private Sub btnDelete_Click()
1616 45 On Error GoTo Err_btnDelete_Click
1617 46
1618 47 DoCmd.DeleteMenuItem acFormBar, acEditMenu, 8, acMenuVer70
1619 48 DoCmd.DeleteMenuItem acFormBar, acEditMenu, 6, acMenuVer70
1620 49
1621 50
1622 51 Exit_btnDelete_Click
1623 52 Exit Sub
1624 53
1625 54 Err_btnDelete_Click
1626 55 MsgBox Err.Description
1627 56 Resume Exit_btnDelete_Click
1628 57
1629 58 End Sub
1630 59 Private Sub btnClose_Click()
1631 60 On Error GoTo Err_btnClose_Click
1632 61
1633 62 DoCmd.Close
1634 63 DoCmd.OpenForm "frmMaintenanceSwitchboard"
1635 64
1636 65 Exit_btnClose_Click
1637 66 Exit Sub
1638 67
1639 68 Err_btnClose_Click
1640 69 MsgBox Err.Description
1641 70 Resume Exit_btnClose_Click
1642 71
1643 72 End Sub
1644 73 End Sub
1645
1646 Form: frmPrepareAndExecuteSoler
1647 Code
1648 1 Attribute VB_Name = "Form_frmPrepareAndExecuteSoler"
1649 2 Attribute VB_Creatable = True
1650 3 Attribute VB_PredeclaredId = True
1651 4 Attribute VB_Exposed = False
1652 5 Option Compare Database
1653 6 Option Explicit
1654 7
1655 8 Private Sub btnAMPL_Click()
1656 9 On Error GoTo Err_btnAMPL_Click
1657 10
1658 11 AmplData
1659 12 Dim stAppName As String
1660 13
1661 14 stAppName = "C:\ToBeRdm\Ampl\win\ampwin.exe"
1662 15 stAppName = "C:\ToBeRdm\Ampl\win\amp.bat"
1663 16 Call Shell(stAppName, 0)
1664 17
1665 18 MsgBox "Enter into the command line: 'include rdm.run'"
1666 19
1667 20 Exit_btnAMPL_Click
1668 21 Exit Sub
1669 22
1670 23 Err_btnAMPL_Click
1671 24 MsgBox Err.Description
1672 25 Resume Exit_btnAMPL_Click
1673 26
1674 27 End Sub
1675 28
1676 29 Private Sub btnAmplPlus_Click()
1677 30 On Error GoTo Err_btnAmplPlus_Click
1678 31
1679 32 AmplData
1680 33 Dim stAppName As String
1681 34
1682 35 stAppName = "C:\AMPLPLUS\AMPLPLUS.EXE rdm.amp"
1683 36 Call Shell(stAppName, 1)
1684 37
1685 38 Exit_btnAmplPlus_Click
1686 39 Exit Sub
1687 40
1688 41 Err_btnAmplPlus_Click
1689 42 MsgBox Err.Description
1690 43
1691 44

```

```

1691 45 Resume Exit_btnAmplPlus_Click
1692 46
1693 47 End Sub
1694 48
1695 49 Private Sub btnReturnPrevious_Click()
1696 50 On Error GoTo Err_btnReturnPrevious_Click
1697 51
1698 52 Dim stDocName As String
1699 53 Dim stLinkCriteria As String
1700 54
1701 55 ' Close the current form
1702 56 DoCmd.Close
1703 57 ' Open specified form
1704 58 stDocName = "frmPreprocessing&ExecutionSwitchboard"
1705 59 DoCmd.OpenForm stDocName, , stLinkCriteria
1706 60
1707 61 Exit_btnReturnPrevious_Click
1708 62 Exit Sub
1709 63
1710 64 Err_btnReturnPrevious_Click
1711 65 MsgBox Err.Description
1712 66 Resume Exit_btnReturnPrevious_Click
1713 67
1714 68 End Sub
1715 69
1716 70 Private Sub Command4_Click()
1717 71 Application.SetOption "Confirm Action Queries", False
1718 72 DoCmd.RunSQL "DELETE FROM AMPL_RESULT;"
1719 73 Application.SetOption "Confirm Action Queries", True
1720 74 DoCmd.SetWarnings (True)
1721 75
1722 76 End Sub
1723 77
1724 78 Private Sub Command7_Click()
1725 79 On Error GoTo Err_Command7_Click
1726 80
1727 81 Dim stDocName As String
1728 82 Dim stLinkCriteria As String
1729 83
1730 84 ' Close the current form
1731 85 DoCmd.Close
1732 86 ' Open specified form
1733 87 stDocName = "frmAnalyzeResult"
1734 88 DoCmd.OpenForm stDocName, , stLinkCriteria
1735 89
1736 90 Exit_Command7_Click
1737 91 Exit Sub
1738 92
1739 93 Err_Command7_Click
1740 94 MsgBox Err.Description
1741 95 Resume Exit_Command7_Click
1742 96
1743 97 End Sub
1744 98
1745 99 Form: frmPreprocessing&ExecutionSwitchboard
1746 Code
1747 1 Attribute VB_Name = "Form_frmPreprocessing&ExecutionSwitchboard"
1748 2 Attribute VB_Creatable = True
1749 3 Attribute VB_PredeclaredId = True
1750 4 Attribute VB_Exposed = False
1751 5 Option Compare Database
1752 6 Option Explicit
1753 7
1754 8 Private Sub btnExt_Click()
1755 9 On Error GoTo Err_btnExt_Click
1756 10
1757 11 DoCmd.Close
1758 12 Exit_btnExt_Click
1759 13 Exit Sub
1760 14
1761 15 Err_btnExt_Click
1762 16 MsgBox Err.Description
1763 17 Resume Exit_btnExt_Click
1764 18
1765 19 End Sub
1766 20
1767 21 Private Sub btnReclassification_Click()
1768 22 On Error GoTo Err_btnReclassification_Click
1769 23
1770 24 Dim stDocName As String
1771 25 Dim stLinkCriteria As String
1772 26
1773 27 ' Close the current form
1774 28 DoCmd.Close
1775 29 ' Open specified form
1776 30 stDocName = "frmReclassification"
1777 31 DoCmd.OpenForm stDocName, , stLinkCriteria
1778 32
1779 33 Exit_btnReclassification_Click
1780 34 Exit Sub
1781 35
1782 36 Err_btnReclassification_Click
1783 37 MsgBox Err.Description
1784 38 Resume Exit_btnReclassification_Click
1785 39
1786 40 End Sub
1787 41
1788 42 Private Sub btnScrubMarineData_Click()
1789 43 On Error GoTo Err_btnScrubMarineData_Click
1790 44
1791 45 Dim stDocName As String
1792 46 Dim stLinkCriteria As String
1793 47
1794 48 ' Close the current form
1795 49 DoCmd.Close
1796 50 ' Open specified form
1797 51 stDocName = "frmScrubMarine"
1798 52 DoCmd.OpenForm stDocName, , stLinkCriteria
1799 53
1800 54 Exit_btnScrubMarineData_Click
1801 55 Exit Sub
1802 56
1803 57 Err_btnScrubMarineData_Click
1804 58 MsgBox Err.Description
1805 59 Resume Exit_btnScrubMarineData_Click
1806 60
1807 61 End Sub
1808 62
1809 63 Private Sub btnSpecialAssignment_Click()
1810 64 On Error GoTo Err_btnSpecialAssignment_Click
1811 65
1812 66
1813 67
1814 68
1815 69

```

```

1821 70 Dim stDocName As String
1822 71 Dim stLinkCriteria As String
1823 72
1824 73 ' Close the current form
1825 74 DoCmd.Close
1826 75 ' Open specified form
1827 76 stDocName = "frmSpecialAssignment"
1828 77 DoCmd.OpenForm stDocName, , stLinkCriteria
1829 78
1830 79 Exit_btnSpecialAssignment_Click
1831 80 Exit Sub
1832 81
1833 82 Err_btnSpecialAssignment_Click
1834 83 MsgBox Err.Description
1835 84 Resume Exit_btnSpecialAssignment_Click
1836 85
1837 86
1838 87
1839 88 End Sub
1840 89
1841 90 Private Sub Command2_Click()
1842 91 On Error GoTo Err_Command2_Click
1843 92
1844 93 Dim stDocName As String
1845 94 Dim stLinkCriteria As String
1846 95
1847 96 ' Close the current form
1848 97 DoCmd.Close
1849 98 ' Open specified form
1850 99 stDocName = "frmClassQuotaPenaltyAndFit"
1851 100 DoCmd.OpenForm stDocName, , stLinkCriteria
1852 101
1853 102 Exit_Command2_Click
1854 103 Exit Sub
1855 104
1856 105 Err_Command2_Click
1857 106 MsgBox Err.Description
1858 107 Resume Exit_Command2_Click
1859 108
1860 109 End Sub
1861 110 Private Sub Command4_Click()
1862 111
1863 112 Application.SetOption "Confirm Action Queries", False
1864 113 DoCmd.SetWarnings (False)
1865 114 DoCmd.RunSQL "DELETE * FROM AMPL_RESULT;"
1866 115 Ampl_Result
1867 116 Application.SetOption "Confirm Action Queries", True
1868 117 DoCmd.SetWarnings (True)
1869 118
1870 119
1871 120 End Sub
1872 121
1873 122 Private Sub Command5_Click()
1874 123 On Error GoTo Err_Command5_Click
1875 124
1876 125 Dim stDocName As String
1877 126 Dim stLinkCriteria As String
1878 127
1879 128 ' Close the current form
1880 129 DoCmd.Close
1881 130 ' Open specified form
1882 131 stDocName = "frmPrepareAndExecuteSolver"
1883 132 DoCmd.OpenForm stDocName, , stLinkCriteria
1884 133
1885 134 Exit_Command5_Click
1886 135 Exit Sub
1887 136
1888 137 Err_Command5_Click
1889 138 MsgBox Err.Description
1890 139 Resume Exit_Command5_Click
1891 140
1892 141 End Sub
1893 142
1894 143 Private Sub Command6_Click()
1895 144 On Error GoTo Err_Command6_Click
1896 145
1897 146 Dim stDocName As String
1898 147 Dim stLinkCriteria As String
1899 148
1900 149 ' Close the current form
1901 150 DoCmd.Close
1902 151 ' Open specified form
1903 152 stDocName = "frmRDM_Main_Switchboard"
1904 153 DoCmd.OpenForm stDocName, , stLinkCriteria
1905 154
1906 155 Exit_Command6_Click
1907 156 Exit Sub
1908 157
1909 158 Err_Command6_Click
1910 159 MsgBox Err.Description
1911 160 Resume Exit_Command6_Click
1912 161
1913 162 End Sub
1914
1915 Form: frmRDM_Main_Switchboard
1916 Code
1917 1 Attribute VB_Name = "Form_frmRDM_Main_Switchboard"
1918 2 Attribute VB_Creatable = True
1919 3 Attribute VB_PredeclaredId = True
1920 4 Attribute VB_Exposed = False
1921 5 Option Compare Database
1922 6 Option Explicit
1923 7 Private Sub Command0_Click()
1924 8
1925 9 On Error GoTo Err_Command0_Click
1926 10
1927 11 Dim stDocName As String
1928 12 Dim stLinkCriteria As String
1929 13
1930 14 ' Close the current form
1931 15 DoCmd.Close
1932 16 ' Open specified form
1933 17 stDocName = "frmImportExportSwitchboard"
1934 18 DoCmd.OpenForm stDocName, , stLinkCriteria
1935 19
1936 20 Exit_Command0_Click
1937 21 Exit Sub
1938
1939 22 Err_Command0_Click
1940 23 MsgBox Err.Description
1941 24 Resume Exit_Command0_Click
1942 25
1943 26 End Sub
1944 27 Private Sub Command2_Click()
1945 28 On Error GoTo Err_Command2_Click
1946 29
1947 30 Dim stDocName As String
1948 31 Dim stLinkCriteria As String
1949 32
1950 33 ' Close the current form

```

```

1951 35 DoCmd.Close
1952 36 ' Open specified form
1953 37 stDocName = "frmMaintenanceSwitchboard"
1954 38 DoCmd.OpenForm stDocName, , stLinkCriteria
1955 39
1956 40 Exit_Command2_Click
1957 41 Exit Sub
1958 42
1959 43 Err_Command2_Click
1960 44 MsgBox Err.Description
1961 45 Resume Exit_Command2_Click
1962 46
1963 47 End Sub
1964 48 Private Sub Command4_Click()
1965 49 On Error GoTo Err_Command4_Click
1966 50
1967 51 Dim stDocName As String
1968 52 Dim stLinkCriteria As String
1969 53
1970 54 ' Close the current form
1971 55 DoCmd.Close
1972 56 ' Open specified form
1973 57 stDocName = "frmPreprocessingExecutionSwitchboard"
1974 58 DoCmd.OpenForm stDocName, , stLinkCriteria
1975 59
1976 60 Exit_Command4_Click
1977 61 Exit Sub
1978 62
1979 63 Err_Command4_Click
1980 64 MsgBox Err.Description
1981 65 Resume Exit_Command4_Click
1982 66
1983 67 End Sub
1984 68 Private Sub Command6_Click()
1985 69 On Error GoTo Err_Command6_Click
1986 70
1987 71 Dim stDocName As String
1988 72 Dim stLinkCriteria As String
1989 73
1990 74 ' Close the current form
1991 75 DoCmd.Close
1992 76 ' Open specified form
1993 77 stDocName = "frmAnalyzeResult"
1994 78 DoCmd.OpenForm stDocName, , stLinkCriteria
1995 79
1996 80 Exit_Command6_Click
1997 81 Exit Sub
1998 82
1999 83 Err_Command6_Click
2000 84 MsgBox Err.Description
2001 85 Resume Exit_Command6_Click
2002 86
2003 87 End Sub
2004 88 Private Sub Command8_Click()
2005 89 On Error GoTo Err_Command8_Click
2006 90
2007 91 Dim stDocName As String
2008 92 Dim stLinkCriteria As String
2009 93
2010 94 ' Close the current form
2011 95 DoCmd.Close
2012 96 ' Open specified form
2013 97 stDocName = "frmGenerateReport"
2014 98 DoCmd.OpenForm stDocName, , stLinkCriteria
2015 99
2016 100 Exit_Command8_Click
2017 101 Exit Sub
2018 102
2019 103 Err_Command8_Click
2020 104 MsgBox Err.Description
2021 105 Resume Exit_Command8_Click
2022 106
2023 107 End Sub
2024 108 Private Sub btnExit_Click()
2025 109 On Error GoTo Err_btnExit_Click
2026 110
2027 111
2028 112 DoCmd.Close
2029 113
2030 114 Exit_btnExit_Click
2031 115 Exit Sub
2032 116
2033 117 Err_btnExit_Click
2034 118 MsgBox Err.Description
2035 119 Resume Exit_btnExit_Click
2036 120
2037 121 End Sub
2038
2039 Form: frmRecClassification
2040 Code
2041 1 Attribute VB_Name = "Form_frmRecClassification"
2042 2 Attribute VB_Creatable = True
2043 3 Attribute VB_PredeclaredId = True
2044 4 Attribute VB_Exposed = False
2045 5 Option Compare Database
2046 6 Option Explicit
2047 7
2048 8 Private Sub cmbClassNumber_AfterUpdate()
2049 9
2050 10 Dim strConvert As String
2051 11
2052 12 Me.txtReportDate = Me.cmbClassNumber.Column(2)
2053 13 Me.txtMOC = Me.cmbClassNumber.Column(4)
2054 14
2055 15 ' Convert the fiscal year to a two digit number for RDS File
2056 16 strConvert = CStr(Me.cmbClassNumber.Column(3))
2057 17 strConvert = Right(strConvert, 2)
2058 18 Me.txtFY = CInt(strConvert)
2059 19
2060 20 Me.txtAssignmentType = "R"
2061 21
2062 22 End Sub
2063 23
2064 24 Private Sub cmbClassNumber_Enter()
2065 25
2066 26 Dim strSQL As String
2067 27
2068 28 ' This query finds the class numbers, report dates and class
2069 29 covering dates associated with the chosen school
2070 30 strSQL = "SELECT BNA_EXTRACT.ClassNumber_PK,
2071 BNA_EXTRACT.ClassOnDate, BNA_EXTRACT.ReportDate,
2072 BNA_EXTRACT.FiscalYear_PK, BNA_EXTRACT.MOC FROM BNA_EXTRACT INNER JOIN
2073 SCH_TGT_MOS ON (BNA_EXTRACT.CourseNumber_PK =
2074 SCH_TGT_MOS.CourseNumber_PK) AND (BNA_EXTRACT.TargetMOS_PK =
2075 SCH_TGT_MOS.TargetMOS_PK) WHERE BNA_EXTRACT.ReportDate
2076 Format(frmRecClassification.txtClassNumber AND SCH_TGT_MOS.AMOS_PK =
2077 Format(frmRecClassification.txtAMOS);"
2078 31 Me.cmbClassNumber.RowSource = strSQL
2079 32
2080 32 End Sub

```

```

2081 33 Private Sub cmbCourseNumber_AfterUpdate()
2082 34 Private Sub cmbCourseNumber_AfterUpdate()
2083 35
2084 36 Me.tbAMOS = Me.cmbCourseNumber.Column(1)
2085 37
2086 38 End Sub
2087 39
2088 40 Private Sub cmbSSN_AfterUpdate()
2089 41
2090 42 Me.tbSQL = Me.cmbSSN.Column(1)
2091 43 Me.tbGradDate = Me.cmbSSN.Column(2)
2092 44
2093 45 End Sub
2094 46
2095 47 Private Sub cmbSSN_Enter()
2096 48
2097 49 Dim strSQL As String
2098 50
2099 51 strSQL = "SELECT DISTINCTROW MARINE.SSN_PK, MARINE.SOI,
2100 MARINE.GradDate FROM MARINE LEFT JOIN ASSIGNMENT ON MARINE.SSN_PK =
2101 ASSIGNMENT.SSN_FK WHERE ((ASSIGNMENT.SSN_FK) Is Null);"
2102 52 Me.cmbSSN.RowSource = strSQL
2103 53
2104 54 End Sub
2105 55
2106 56 Private Sub cmbSSNFind_AfterUpdate()
2107 57
2108 58 Dim R As Recordset
2109 59 Set R = Me.RecordsetClone
2110 60 R.FindFirst "[SSN_FK] = " & Chr(34) & Me.cmbSSNFind & Chr(34)
2111 61 Me.Bookmark = R.Bookmark
2112 62 Me.cmbSSNFind = Null
2113 63 Me.cmbSSN.SetFocus
2114 64
2115 65 End Sub
2116 66
2117 67 Private Sub Delete_Click()
2118 68 On Error GoTo Err_Delete_Click
2119 69
2120 70
2121 71 DoCmd.DoMenuItem acFormBar, acEditMenu, 8, , acMenuVer70
2122 72 DoCmd.DoMenuItem acFormBar, acEditMenu, 8, , acMenuVer70
2123 73
2124 74 Exit_Delete_Click
2125 75 Exit Sub
2126 76
2127 77 Err_Delete_Click:
2128 78 MsgBox Err.Description
2129 79 Resume Exit_Delete_Click
2130 80
2131 81 End Sub
2132 82
2133 83 Private Sub Close_Click()
2134 84 On Error GoTo Err_Close_Click
2135 85
2136 86 Dim stDocName As String
2137 87 Dim stLinkCriteria As String
2138 88
2139 89 ' Close current form
2140 90 DoCmd.Close
2141 91
2142 92 ' Open specified form
2143 93 stDocName = "frmProcessing&ExecutionSwitchboard"
2144 94 DoCmd.OpenForm stDocName, , stLinkCriteria
2145 95
2146 96 Exit_Close_Click
2147 97 Exit Sub
2148 98
2149 99 Err_Close_Click:
2150 100 MsgBox Err.Description
2151 101 Resume Exit_Close_Click
2152 102
2153 103 End Sub
2154 104
2155 105
2156 106 Form: frmSchoolAssignments
2157 107 Code
2158 108 1 Attribute VB_Name = "Form_frmSchoolAssignments"
2159 109 2 Attribute VB_Creatable = True
2160 110 3 Attribute VB_PredeclaredId = True
2161 111 4 Attribute VB_Exposed = False
2162 112 5 Option Compare Database
2163 113 6 Option Explicit
2164 114 7 Private Sub Close_Click()
2165 115 8 On Error GoTo Err_Close_Click
2166 116 9
2167 117 10 Dim stDocName As String
2168 118 11 Dim stLinkCriteria As String
2169 119 12 Dim db1 As Database
2170 120 13
2171 121 14 Set db1 = CurrentDb()
2172 122 15
2173 123 16 ' Close the current form
2174 124 17 DoCmd.Close
2175 125 18 ' Open specified form
2176 126 19 stDocName = "frmAnalyzeResult"
2177 127 20 DoCmd.OpenForm stDocName, , stLinkCriteria
2178 128 21 Form!frmAnalyzeResult.SetFocus
2179 129 22
2180 130 23
2181 131 24 Exit_Close_Click
2182 132 25 Exit Sub
2183 133 26
2184 134 27 Err_Close_Click:
2185 135 28 MsgBox Err.Description
2186 136 29 Resume Exit_Close_Click
2187 137 30
2188 138 31 End Sub
2189 139 32
2190 140 33 Private Sub Form_Current()
2191 141 34
2192 142 35 Dim strSQL As String
2193 143 36 Dim rec As Recordset
2194 144 37 Dim db1 As Database
2195 145 38
2196 146 39 Set db1 = CurrentDb()
2197 147 40
2198 148 41 ' Calculates the total quota, and number of schools for the run.
2199 149 42 strSQL = "SELECT Sum(SumOfQuota) AS TotalQuota, Count(SumOfQuota) AS
2200 NumberOfSchools FROM qryTotalQuotaForRun;"
2201 150 43 Set rec = db1.OpenRecordset(strSQL, dbOpenSnapshot)
2202 151 44 Me.tbTotalQuota = rec.TotalQuota
2203 152 45 Me.tbNumberOfSchools = rec.NumberOfSchools
2204 153 46
2205 154 47 ' Calculates the total number of assignments for the run.
2206 155 48 strSQL = "SELECT Sum(CountOfAssignments) AS TotalAssigned FROM
2207 qryTotalQuotaForRun;"
2208 156 49 Set rec = db1.OpenRecordset(strSQL, dbOpenSnapshot)
2209 157 50 Me.tbTotalAssigned = rec.TotalAssigned
2210 158 51
2211 159 52 Me.tbTotalPercentFill = Int((Me.tbTotalAssigned / Me.tbTotalQuota)
2212 160 53
2213 161 54
2214 162 55 End Sub
2215 163 56
2216 164 57 Form: frmSchools
2217 165 58 Code
2218 166 59 1 Attribute VB_Name = "Form_frmSchools"
2219 167 60 2 Attribute VB_Creatable = True
2220 168 61 3 Attribute VB_PredeclaredId = True
2221 169 62 4 Attribute VB_Exposed = False
2222 170 63 5 Option Compare Database
2223 171 64 6 Option Explicit
2224 172 65 7
2225 173 66 8 Private Sub AMOS_PK_AfterUpdate()
2226 174 67
2227 175 68 Dim rec As Recordset
2228 176 69 Dim db1 As Database
2229 177 70 Dim strSQL As String
2230 178 71
2231 179 72 ' Update the drop down list for AMOS
2232 180 73 strSQL = "SELECT TargetMOS_PK FROM
2233 181 74 qryListUnusedTargetMOSFromTARGET_MOS WHERE TCourseNumber_PK = " &
2234 182 75 Me.AMOS_PK.RowSource & strSQL
2235 183 76 Me.lstTargetMOS.RowSource = strSQL
2236 184 77
2237 185 78 ' Update the AMOS_FK in the child relation SCH_TGT_MOS
2238 186 79 ' Form!frmSchools.subfrmSchools.AMOS_FK.Value = Me.AMOS_PK.Value
2239 187 80
2240 188 81 ' Update the TargetMOS_FK in the child relation SCH_TGT_MOS
2241 189 82 ' Set db1 = CurrentDb()
2242 190 83 ' Set rec = db1.OpenRecordset(strSQL, dbOpenSnapshot)
2243 191 84 ' Form!frmSchools.subfrmSchools.TargetMOS_FK.Value = rec.TargetMOS_PK
2244 192 85
2245 193 86 ' Input the correct MCC value for the selected course
2246 194 87 strSQL = "SELECT MCC FROM TARGET_MOS WHERE TCourseNumber_PK = " &
2247 195 88 Me.SCourseNumber_PK.Value & ""
2248 196 89 Set db1 = CurrentDb()
2249 197 90 Set rec = db1.OpenRecordset(strSQL, dbOpenSnapshot)
2250 198 91 Me.MCC.Value = rec!MCC
2251 199 92
2252 200 93 ' Update the TCourseNumber_FK in the child relation SCH_TGT_MOS
2253 201 94 ' Form!frmSchools.subfrmSchools.TCourseNumber_FK.Value =
2254 202 95 Me.SCourseNumber_PK.Value
2255 203 96
2256 204 97 End Sub
2257 205 98
2258 206 99 Private Sub btnClear_Click()
2259 207 100
2260 208 101 Dim db1 As Database
2261 209 102 Dim strSQL As String
2262 210 103 Dim i As Integer
2263 211 104 Dim rec As Recordset
2264 212 105
2265 213 106 Set db1 = CurrentDb()
2266 214 107
2267 215 108 strSQL = "SELECT DISTINCTROW TARGET_MOS.TCourseNumber_PK FROM
2268 216 109 TARGET_MOS LEFT JOIN SCH_TGT_MOS ON TARGET_MOS.TargetMOS_PK =
2269 217 110 SCH_TGT_MOS.TargetMOS_FK AND TARGET_MOS.TCourseNumber_PK =
2270 218 111 SCH_TGT_MOS.TCourseNumber_FK WHERE (((SCH_TGT_MOS.TargetMOS_FK) Is
2271 219 112 Null) AND (SCH_TGT_MOS.TCourseNumber_FK) Is Null) ORDER BY
2272 220 113 'TCourseNumber_PK;"
2273 221 114 Set rec = db1.OpenRecordset(strSQL, dbOpenDynaset)
2274 222 115
2275 223 116 If rec.EOF = False Then
2276 224 117 rec.MoveNext
2277 225 118 rec.MoveNext
2278 226 119 End If
2279 227 120
2280 228 121 ' Clear unused course numbers from the target_mos table
2281 229 122
2282 230 123 If rec.RecordCount > 0 Then
2283 231 124 For i = 1 To rec.RecordCount
2284 232 125 ' Deletes the action query confirmation message
2285 233 126 Application.SetOption "Confirm Action Queries", False
2286 234 127 DoCmd.SetWarnings (False)
2287 235 128
2288 236 129 strSQL = "DELETE * FROM TARGET_MOS WHERE
2289 237 130 TARGET_MOS.TCourseNumber_PK = " & rec!TCourseNumber_PK & ""
2290 238 131 db1.Execute strSQL
2291 239 132
2292 240 133 ' Enables the action query confirmation message
2293 241 134 Application.SetOption "Confirm Action Queries", True
2294 242 135 DoCmd.SetWarnings (True)
2295 243 136 rec.MoveNext
2296 244 137 Next i
2297 245 138 End If
2298 246 139
2299 247 140 ' Update the Course Number List
2300 248 141 strSQL = "SELECT DISTINCTROW TARGET_MOS.TCourseNumber_PK FROM
2301 249 142 TARGET_MOS LEFT JOIN SCH_TGT_MOS ON TARGET_MOS.TargetMOS_PK =
2302 250 143 SCH_TGT_MOS.TargetMOS_FK AND TARGET_MOS.TCourseNumber_PK =
2303 251 144 SCH_TGT_MOS.TCourseNumber_FK WHERE (((SCH_TGT_MOS.TargetMOS_FK) Is
2304 252 145 Null) AND (SCH_TGT_MOS.TCourseNumber_FK) Is Null) ORDER BY
2305 253 146 'TCourseNumber_PK;"
2306 254 147 Me.SCourseNumber_PK.RowSource = strSQL
2307 255 148
2308 256 149 End Sub
2309 257 150
2310 258 151 Private Sub btnResetAllPenalties_Click()
2311 259 152
2312 260 153 ' Disables the action query confirmation message
2313 261 154 Application.SetOption "Confirm Action Queries", False
2314 262 155 DoCmd.SetWarnings (False)
2315 263 156
2316 264 157 DoCmd.RunSQL "UPDATE SCHOOL SET PenaltyFactor = 24;"
2317 265 158
2318 266 159 ' Enables the action query confirmation message
2319 267 160 Application.SetOption "Confirm Action Queries", True
2320 268 161 DoCmd.SetWarnings (True)
2321 269 162
2322 270 163 cmbPenaltyView = "1x"
2323 271 164
2324 272 165 End Sub
2325 273 166
2326 274 167 Private Sub cmbCourseNumberFind_AfterUpdate()
2327 275 168
2328 276 169 Dim R As Recordset
2329 277 170 Set R = Me.RecordsetClone
2330 278 171 R.FindFirst "[TCourseNumber_PK] = " & Chr(34) &
2331 279 172 Me.cmbCourseNumberFind & Chr(34)
2332 280 173 Me.Bookmark = R.Bookmark
2333 281 174 Me.cmbCourseNumberFind = Null
2334 282 175 Me.SCourseNumber_PK.SetFocus
2335 283 176
2336 284 177 End Sub
2337 285 178
2338 286 179 Private Sub cmbPenaltyView_AfterUpdate()
2339 287 180

```

```

2340 108 Select Case cmbPenaltyView
2341 109 Case "4x"
2342 110 bPenaltyFactor = 4 * 24
2343 111 Case "3x"
2344 112 bPenaltyFactor = 3 * 24
2345 113 Case "2x"
2346 114 bPenaltyFactor = 2 * 24
2347 115 Case "1x"
2348 116 bPenaltyFactor = 1 * 24
2349 117 Case "1/2x"
2350 118 bPenaltyFactor = 24 / 2
2351 119 Case "1/3x"
2352 120 bPenaltyFactor = 24 / 3
2353 121 Case "1/4x"
2354 122 bPenaltyFactor = 24 / 4
2355 123 End Select
2356 124
2357 125 ' Save the update
2358 126 DoCmd.DoMenuItem acFormBar, acRecordsMenu, acSaveRecord,
2359 127
2360 128 End Sub
2361 129
2362 130 Private Sub Command14_Click()
2363 131
2364 132 Dim strSQL As String
2365 133
2366 134 ' Disables the action query confirmation message
2367 135 Application.SetOption "Confirm Action Queries", False
2368 136 DoCmd.SetWarnings (False)
2369 137
2370 138 DoCmd.RunSQL "INSERT INTO TARGET_MOS (TCourseNumber_PK,
2371 TargetMOS_PK, NICC) SELECT DISTINCT BNA_EXTRACT.CourseNumber_PK,
2372 BNA_EXTRACT.TargetMOS_PK, BNA_EXTRACT.NICC FROM BNA_EXTRACT;"
2373 139
2374 140 ' Enables the action query confirmation message
2375 141 Application.SetOption "Confirm Action Queries", True
2376 142 DoCmd.SetWarnings (True)
2377 143
2378 144 ' Update the Course Number List
2379 145 strSQL = "SELECT DISTINCTROW TARGET_MOS.TCourseNumber_PK FROM
2380 TARGET_MOS LEFT JOIN SCH_TGT_MOS ON TARGET_MOS.TargetMOS_PK =
2381 SCH_TGT_MOS.TargetMOS_PK AND TARGET_MOS.TCourseNumber_PK =
2382 SCH_TGT_MOS.TCourseNumber_PK WHERE ((SCH_TGT_MOS.TargetMOS_PK) Is
2383 Null)AND((SCH_TGT_MOS.TCourseNumber_PK) Is Null)ORDER BY
2384 TCourseNumber_PK;"
2385 146 Me.SCourseNumber_PK.RowSource = strSQL
2386 147
2387 148 End Sub
2388 149
2389 150
2390 151 Private Sub Command36_Click()
2391 152 Dim frm As Form, ctf As Control
2392 153 Dim varTrim As Variant, int As Integer
2393 154 Dim strSQL As String
2394 155
2395 156 ' Ensures the current record is save to the SCHOOL table
2396 157 DoCmd.DoMenuItem acFormBar, acRecordsMenu, acSaveRecord,
2397 158
2398 159 ' This code enters the selected Target MOS's into the SCH_TGT_MOS
2399 160 Set frm = Form!frmSchoole
2400 161 Set ctf = frm!txtTargetMOS_FK
2401 162 For Each varTrim In ctf.ItemsSelected
2402 163 For int = 0 To ctf.ColumnCount - 1
2403 164
2404 165 ' This puts the value found in the list box into a text box,
2405 166 making it readable by the query
2406 167 Form!frmSchoole!TargetValue.Value = ctf.Column(int, varTrim)
2407 168
2408 169 ' Disables the action query confirmation message
2409 170 Application.SetOption "Confirm Action Queries", False
2410 171
2411 172 ' Enters the value into the SCH_TGT_MOS table
2412 173 DoCmd.OpenQuery "qryUpdateSCH_TGT_MOS"
2413 174
2414 175 ' Enables the action query confirmation message
2415 176 Application.SetOption "Confirm Action Queries", True
2416 177
2417 178 Next int
2418 179 Next varTrim
2419 180
2420 181 ' Update the Target MOS list
2421 182 strSQL = "SELECT [TargetMOS_FK] FROM [SCH_TGT_MOS] WHERE
2422 [SCourseNumber_PK] = " & Me.SCourseNumber_PK.Value & " " &
2423 "AND AMOS_PK = " & Me.AMOS_PK.Value & " " &
2424 "Form!frmSchoole!ListTargetMOS_FK_History.RowSource = strSQL
2425 183
2426 184 ' Update the Potential Target MOS list
2427 185 strSQL = "SELECT TargetMOS_PK FROM
2428 qryListUsedTargetMOSFromTARGET_MOS WHERE TCourseNumber_PK = " &
2429 Me.ListTargetMOS_FK.RowSource = strSQL
2430 186
2431 187 ' Update the Course Number List
2432 188 strSQL = "SELECT DISTINCTROW TARGET_MOS.TCourseNumber_PK FROM
2433 TARGET_MOS LEFT JOIN SCH_TGT_MOS ON TARGET_MOS.TargetMOS_PK =
2434 SCH_TGT_MOS.TargetMOS_PK WHERE ((SCH_TGT_MOS.TargetMOS_PK) Is
2435 Null)AND((SCH_TGT_MOS.TCourseNumber_PK) Is Null)ORDER BY
2436 TCourseNumber_PK;"
2437 189 Me.SCourseNumber_PK.RowSource = strSQL
2438 190
2439 191 ' Update time stamp
2440 192 [Sch_TimeStamp] = Now
2441 193 End Sub
2442 194
2443 195 Private Sub Command40_Click()
2444 196 Dim frm As Form, ctf As Control
2445 200 Dim varTrim As Variant, int As Integer
2446 201 Dim strSQL As String
2447 202
2448 203 ' This code Deletes the selected Target MOS's and its associated
2449 204 entries from the SCH_TGT_MOS table
2450 205 Set frm = Form!frmSchoole
2451 206 Set ctf = frm!ListTargetMOS_FK_History
2452 207 For Each varTrim In ctf.ItemsSelected
2453 208 For int = 0 To ctf.ColumnCount - 1
2454 209
2455 210 ' This puts the value found in the list box into a text box,
2456 211 making it readable by the query
2457 212 Form!frmSchoole!TargetValue.Value = ctf.Column(int, varTrim)
2458 213
2459 214 ' Disables the action query confirmation message
2460 215 Application.SetOption "Confirm Action Queries", False
2461 216
2462 217 ' Deletes the record associated with the specified value from
2463 218 the SCH_TGT_MOS table
2464 219 DoCmd.OpenQuery "qryDeleteSCH_TGT_MOS"
2465 220
2466 221 ' Enables the action query confirmation message
2467 222 Application.SetOption "Confirm Action Queries", True
2468 223
2469 224 Next int
2470 225
2471 226 Next varTrim
2472 227
2473 228 ' Update the Target MOS list
2474 229 strSQL = "SELECT [TargetMOS_FK] FROM [SCH_TGT_MOS] WHERE
2475 230 [SCourseNumber_PK] = " & Me.SCourseNumber_PK.Value & " " &
2476 231 "AND AMOS_PK = " & Me.AMOS_PK.Value & " " &
2477 232 "Form!frmSchoole!ListTargetMOS_FK_History.RowSource = strSQL
2478 233
2479 234 ' Update the Potential Target MOS list
2480 235 strSQL = "SELECT TargetMOS_PK FROM
2481 236 qryListUsedTargetMOSFromTARGET_MOS WHERE TCourseNumber_PK = " &
2482 237 Me.ListTargetMOS_FK.RowSource = strSQL
2483 238
2484 239 ' Update the Course Number List
2485 240 strSQL = "SELECT DISTINCTROW TARGET_MOS.TCourseNumber_PK FROM
2486 241 TARGET_MOS LEFT JOIN SCH_TGT_MOS ON TARGET_MOS.TargetMOS_PK =
2487 242 SCH_TGT_MOS.TargetMOS_PK AND TARGET_MOS.TCourseNumber_PK =
2488 243 SCH_TGT_MOS.TCourseNumber_PK WHERE ((SCH_TGT_MOS.TargetMOS_PK) Is
2489 244 Null)AND((SCH_TGT_MOS.TCourseNumber_PK) Is Null)ORDER BY
2490 245 TCourseNumber_PK;"
2491 246 Me.SCourseNumber_PK.RowSource = strSQL
2492 247
2493 248 ' Forces user to enter date for course number, before the Assigned
2494 249 If IsNull(Me.SCourseNumber_PK.Value) Then
2495 250 Me!AMOS_PK.Enabled = False
2496 251
2497 252 ' Allows the user to enter an assigned MOS
2498 253 Else
2499 254 Me!AMOS_PK.Enabled = True
2500 255 End If
2501 256
2502 257 ' Prevents the user from accidentally changing a course number
2503 258 If IsNull(Me.SCourseNumber_PK) Then
2504 259 Me!SCourseNumber_PK.Locked = False
2505 260 Else
2506 261 Me!SCourseNumber_PK.Locked = True
2507 262 End If
2508 263
2509 264 ' Prevents the user from accidentally changing an AMOS
2510 265 If IsNull(Me.AMOS_PK) Then
2511 266 Me!AMOS_PK.Locked = False
2512 267 Else
2513 268 Me!AMOS_PK.Locked = True
2514 269 End If
2515 270
2516 271 ' Update the penalty factor view
2517 272 Select Case cmbPenaltyView
2518 273 Case 06
2519 274 cmbPenaltyView = "4x"
2520 275
2521 276 Case 72
2522 277 cmbPenaltyView = "3x"
2523 278
2524 279 Case 48
2525 280 cmbPenaltyView = "2x"
2526 281
2527 282 Case 24
2528 283 cmbPenaltyView = "1x"
2529 284
2530 285 Case 12
2531 286 cmbPenaltyView = "1/2x"
2532 287
2533 288 Case 8
2534 289 cmbPenaltyView = "1/3x"
2535 290
2536 291 Case 6
2537 292 cmbPenaltyView = "1/4x"
2538 293 End Select
2539 294
2540 295 End Sub
2541 296
2542 297 Private Sub btnDelete_Click()
2543 298 Dim rec As Recordset
2544 299 Dim db1 As Database
2545 300 Dim strSQL As String
2546 301
2547 302 ' Update the drop down list for AMOS
2548 303 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
2549 304 = " & Me.SCourseNumber_PK.Value & " " &
2550 305 "Me.AMOS_PK.RowSource = strSQL
2551 306 Me.ListTargetMOS_FK.RowSource = strSQL
2552 307
2553 308 ' Allows the user to enter an assigned MOS
2554 309 Me!AMOS_PK.Enabled = True
2555 310
2556 311 End Sub
2557 312
2558 313 Private Sub btnDelete_Click()
2559 314 Dim rec As Recordset
2560 315 Dim db1 As Database
2561 316 Dim strSQL As String
2562 317
2563 318 ' Update the drop down list for AMOS
2564 319 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
2565 320 = " & Me.SCourseNumber_PK.Value & " " &
2566 321 "Me.AMOS_PK.RowSource = strSQL
2567 322 Me.ListTargetMOS_FK.RowSource = strSQL
2568 323
2569 324 ' Allows the user to enter an assigned MOS
2570 325 Me!AMOS_PK.Enabled = True
2571 326
2572 327 End Sub
2573 328
2574 329 Private Sub btnDelete_Click()
2575 330 Dim rec As Recordset
2576 331 Dim db1 As Database
2577 332 Dim strSQL As String
2578 333
2579 334 ' Update the drop down list for AMOS
2580 335 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
2581 336 = " & Me.SCourseNumber_PK.Value & " " &
2582 337 "Me.AMOS_PK.RowSource = strSQL
2583 338 Me.ListTargetMOS_FK.RowSource = strSQL
2584 339
2585 340 ' Allows the user to enter an assigned MOS
2586 341 Me!AMOS_PK.Enabled = True
2587 342
2588 343 End Sub
2589 344
2590 345 Private Sub btnDelete_Click()
2591 346 Dim rec As Recordset
2592 347 Dim db1 As Database
2593 348 Dim strSQL As String
2594 349
2595 350 ' Update the drop down list for AMOS
2596 351 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
2597 352 = " & Me.SCourseNumber_PK.Value & " " &
2598 353 "Me.AMOS_PK.RowSource = strSQL
2599 354 Me.ListTargetMOS_FK.RowSource = strSQL
2600 355
2601 356 ' Allows the user to enter an assigned MOS
2602 357 Me!AMOS_PK.Enabled = True
2603 358
2604 359 End Sub
2605 360
2606 361 Private Sub btnDelete_Click()
2607 362 Dim rec As Recordset
2608 363 Dim db1 As Database
2609 364 Dim strSQL As String
2610 365
2611 366 ' Update the drop down list for AMOS
2612 367 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
2613 368 = " & Me.SCourseNumber_PK.Value & " " &
2614 369 "Me.AMOS_PK.RowSource = strSQL
2615 370 Me.ListTargetMOS_FK.RowSource = strSQL
2616 371
2617 372 ' Allows the user to enter an assigned MOS
2618 373 Me!AMOS_PK.Enabled = True
2619 374
2620 375 End Sub
2621 376
2622 377 Private Sub btnDelete_Click()
2623 378 Dim rec As Recordset
2624 379 Dim db1 As Database
2625 380 Dim strSQL As String
2626 381
2627 382 ' Update the drop down list for AMOS
2628 383 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
2629 384 = " & Me.SCourseNumber_PK.Value & " " &
2630 385 "Me.AMOS_PK.RowSource = strSQL
2631 386 Me.ListTargetMOS_FK.RowSource = strSQL
2632 387
2633 388 ' Allows the user to enter an assigned MOS
2634 389 Me!AMOS_PK.Enabled = True
2635 390
2636 391 End Sub
2637 392
2638 393 Private Sub btnDelete_Click()
2639 394 Dim rec As Recordset
2640 395 Dim db1 As Database
2641 396 Dim strSQL As String
2642 397
2643 398 ' Update the drop down list for AMOS
2644 399 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
2645 400 = " & Me.SCourseNumber_PK.Value & " " &
2646 401 "Me.AMOS_PK.RowSource = strSQL
2647 402 Me.ListTargetMOS_FK.RowSource = strSQL
2648 403
2649 404 ' Allows the user to enter an assigned MOS
2650 405 Me!AMOS_PK.Enabled = True
2651 406
2652 407 End Sub
2653 408
2654 409 Private Sub btnDelete_Click()
2655 410 Dim rec As Recordset
2656 411 Dim db1 As Database
2657 412 Dim strSQL As String
2658 413
2659 414 ' Update the drop down list for AMOS
2660 415 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
2661 416 = " & Me.SCourseNumber_PK.Value & " " &
2662 417 "Me.AMOS_PK.RowSource = strSQL
2663 418 Me.ListTargetMOS_FK.RowSource = strSQL
2664 419
2665 420 ' Allows the user to enter an assigned MOS
2666 421 Me!AMOS_PK.Enabled = True
2667 422
2668 423 End Sub
2669 424
2670 425 Private Sub btnDelete_Click()
2671 426 Dim rec As Recordset
2672 427 Dim db1 As Database
2673 428 Dim strSQL As String
2674 429
2675 430 ' Update the drop down list for AMOS
2676 431 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
2677 432 = " & Me.SCourseNumber_PK.Value & " " &
2678 433 "Me.AMOS_PK.RowSource = strSQL
2679 434 Me.ListTargetMOS_FK.RowSource = strSQL
2680 435
2681 436 ' Allows the user to enter an assigned MOS
2682 437 Me!AMOS_PK.Enabled = True
2683 438
2684 439 End Sub
2685 440
2686 441 Private Sub btnDelete_Click()
2687 442 Dim rec As Recordset
2688 443 Dim db1 As Database
2689 444 Dim strSQL As String
2690 445
2691 446 ' Update the drop down list for AMOS
2692 447 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
2693 448 = " & Me.SCourseNumber_PK.Value & " " &
2694 449 "Me.AMOS_PK.RowSource = strSQL
2695 450 Me.ListTargetMOS_FK.RowSource = strSQL
2696 451
2697 452 ' Allows the user to enter an assigned MOS
2698 453 Me!AMOS_PK.Enabled = True
2699 454
2700 455 End Sub
2701 456
2702 457 Private Sub btnDelete_Click()
2703 458 Dim rec As Recordset
2704 459 Dim db1 As Database
2705 460 Dim strSQL As String
2706 461
2707 462 ' Update the drop down list for AMOS
2708 463 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
2709 464 = " & Me.SCourseNumber_PK.Value & " " &
2710 465 "Me.AMOS_PK.RowSource = strSQL
2711 466 Me.ListTargetMOS_FK.RowSource = strSQL
2712 467
2713 468 ' Allows the user to enter an assigned MOS
2714 469 Me!AMOS_PK.Enabled = True
2715 470
2716 471 End Sub
2717 472
2718 473 Private Sub btnDelete_Click()
2719 474 Dim rec As Recordset
2720 475 Dim db1 As Database
2721 476 Dim strSQL As String
2722 477
2723 478 ' Update the drop down list for AMOS
2724 479 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
2725 480 = " & Me.SCourseNumber_PK.Value & " " &
2726 481 "Me.AMOS_PK.RowSource = strSQL
2727 482 Me.ListTargetMOS_FK.RowSource = strSQL
2728 483
2729 484 ' Allows the user to enter an assigned MOS
2730 485 Me!AMOS_PK.Enabled = True
2731 486
2732 487 End Sub
2733 488
2734 489 Private Sub btnDelete_Click()
2735 490 Dim rec As Recordset
2736 491 Dim db1 As Database
2737 492 Dim strSQL As String
2738 493
2739 494 ' Update the drop down list for AMOS
2740 495 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
2741 496 = " & Me.SCourseNumber_PK.Value & " " &
2742 497 "Me.AMOS_PK.RowSource = strSQL
2743 498 Me.ListTargetMOS_FK.RowSource = strSQL
2744 499
2745 500 ' Allows the user to enter an assigned MOS
2746 501 Me!AMOS_PK.Enabled = True
2747 502
2748 503 End Sub
2749 504
2750 505 Private Sub btnDelete_Click()
2751 506 Dim rec As Recordset
2752 507 Dim db1 As Database
2753 508 Dim strSQL As String
2754 509
2755 510 ' Update the drop down list for AMOS
2756 511 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
2757 512 = " & Me.SCourseNumber_PK.Value & " " &
2758 513 "Me.AMOS_PK.RowSource = strSQL
2759 514 Me.ListTargetMOS_FK.RowSource = strSQL
2760 515
2761 516 ' Allows the user to enter an assigned MOS
2762 517 Me!AMOS_PK.Enabled = True
2763 518
2764 519 End Sub
2765 520
2766 521 Private Sub btnDelete_Click()
2767 522 Dim rec As Recordset
2768 523 Dim db1 As Database
2769 524 Dim strSQL As String
2770 525
2771 526 ' Update the drop down list for AMOS
2772 527 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
2773 528 = " & Me.SCourseNumber_PK.Value & " " &
2774 529 "Me.AMOS_PK.RowSource = strSQL
2775 530 Me.ListTargetMOS_FK.RowSource = strSQL
2776 531
2777 532 ' Allows the user to enter an assigned MOS
2778 533 Me!AMOS_PK.Enabled = True
2779 534
2780 535 End Sub
2781 536
2782 537 Private Sub btnDelete_Click()
2783 538 Dim rec As Recordset
2784 539 Dim db1 As Database
2785 540 Dim strSQL As String
2786 541
2787 542 ' Update the drop down list for AMOS
2788 543 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
2789 544 = " & Me.SCourseNumber_PK.Value & " " &
2790 545 "Me.AMOS_PK.RowSource = strSQL
2791 546 Me.ListTargetMOS_FK.RowSource = strSQL
2792 547
2793 548 ' Allows the user to enter an assigned MOS
2794 549 Me!AMOS_PK.Enabled = True
2795 550
2796 551 End Sub
2797 552
2798 553 Private Sub btnDelete_Click()
2799 554 Dim rec As Recordset
2800 555 Dim db1 As Database
2801 556 Dim strSQL As String
2802 557
2803 558 ' Update the drop down list for AMOS
2804 559 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
2805 560 = " & Me.SCourseNumber_PK.Value & " " &
2806 561 "Me.AMOS_PK.RowSource = strSQL
2807 562 Me.ListTargetMOS_FK.RowSource = strSQL
2808 563
2809 564 ' Allows the user to enter an assigned MOS
2810 565 Me!AMOS_PK.Enabled = True
2811 566
2812 567 End Sub
2813 568
2814 569 Private Sub btnDelete_Click()
2815 570 Dim rec As Recordset
2816 571 Dim db1 As Database
2817 572 Dim strSQL As String
2818 573
2819 574 ' Update the drop down list for AMOS
2820 575 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
2821 576 = " & Me.SCourseNumber_PK.Value & " " &
2822 577 "Me.AMOS_PK.RowSource = strSQL
2823 578 Me.ListTargetMOS_FK.RowSource = strSQL
2824 579
2825 580 ' Allows the user to enter an assigned MOS
2826 581 Me!AMOS_PK.Enabled = True
2827 582
2828 583 End Sub
2829 584
2830 585 Private Sub btnDelete_Click()
2831 586 Dim rec As Recordset
2832 587 Dim db1 As Database
2833 588 Dim strSQL As String
2834 589
2835 590 ' Update the drop down list for AMOS
2836 591 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
2837 592 = " & Me.SCourseNumber_PK.Value & " " &
2838 593 "Me.AMOS_PK.RowSource = strSQL
2839 594 Me.ListTargetMOS_FK.RowSource = strSQL
2840 595
2841 596 ' Allows the user to enter an assigned MOS
2842 597 Me!AMOS_PK.Enabled = True
2843 598
2844 599 End Sub
2845 600
2846 601 Private Sub btnDelete_Click()
2847 602 Dim rec As Recordset
2848 603 Dim db1 As Database
2849 604 Dim strSQL As String
2850 605
2851 606 ' Update the drop down list for AMOS
2852 607 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
2853 608 = " & Me.SCourseNumber_PK.Value & " " &
2854 609 "Me.AMOS_PK.RowSource = strSQL
2855 610 Me.ListTargetMOS_FK.RowSource = strSQL
2856 611
2857 612 ' Allows the user to enter an assigned MOS
2858 613 Me!AMOS_PK.Enabled = True
2859 614
2860 615 End Sub
2861 616
2862 617 Private Sub btnDelete_Click()
2863 618 Dim rec As Recordset
2864 619 Dim db1 As Database
2865 620 Dim strSQL As String
2866 621
2867 622 ' Update the drop down list for AMOS
2868 623 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
2869 624 = " & Me.SCourseNumber_PK.Value & " " &
2870 625 "Me.AMOS_PK.RowSource = strSQL
2871 626 Me.ListTargetMOS_FK.RowSource = strSQL
2872 627
2873 628 ' Allows the user to enter an assigned MOS
2874 629 Me!AMOS_PK.Enabled = True
2875 630
2876 631 End Sub
2877 632
2878 633 Private Sub btnDelete_Click()
2879 634 Dim rec As Recordset
2880 635 Dim db1 As Database
2881 636 Dim strSQL As String
2882 637
2883 638 ' Update the drop down list for AMOS
2884 639 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
2885 640 = " & Me.SCourseNumber_PK.Value & " " &
2886 641 "Me.AMOS_PK.RowSource = strSQL
2887 642 Me.ListTargetMOS_FK.RowSource = strSQL
2888 643
2889 644 ' Allows the user to enter an assigned MOS
2890 645 Me!AMOS_PK.Enabled = True
2891 646
2892 647 End Sub
2893 648
2894 649 Private Sub btnDelete_Click()
2895 650 Dim rec As Recordset
2896 651 Dim db1 As Database
2897 652 Dim strSQL As String
2898 653
2899 654 ' Update the drop down list for AMOS
2900 655 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
2901 656 = " & Me.SCourseNumber_PK.Value & " " &
2902 657 "Me.AMOS_PK.RowSource = strSQL
2903 658 Me.ListTargetMOS_FK.RowSource = strSQL
2904 659
2905 660 ' Allows the user to enter an assigned MOS
2906 661 Me!AMOS_PK.Enabled = True
2907 662
2908 663 End Sub
2909 664
2910 665 Private Sub btnDelete_Click()
2911 666 Dim rec As Recordset
2912 667 Dim db1 As Database
2913 668 Dim strSQL As String
2914 669
2915 670 ' Update the drop down list for AMOS
2916 671 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
2917 672 = " & Me.SCourseNumber_PK.Value & " " &
2918 673 "Me.AMOS_PK.RowSource = strSQL
2919 674 Me.ListTargetMOS_FK.RowSource = strSQL
2920 675
2921 676 ' Allows the user to enter an assigned MOS
2922 677 Me!AMOS_PK.Enabled = True
2923 678
2924 679 End Sub
2925 680
2926 681 Private Sub btnDelete_Click()
2927 682 Dim rec As Recordset
2928 683 Dim db1 As Database
2929 684 Dim strSQL As String
2930 685
2931 686 ' Update the drop down list for AMOS
2932 687 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
2933 688 = " & Me.SCourseNumber_PK.Value & " " &
2934 689 "Me.AMOS_PK.RowSource = strSQL
2935 690 Me.ListTargetMOS_FK.RowSource = strSQL
2936 691
2937 692 ' Allows the user to enter an assigned MOS
2938 693 Me!AMOS_PK.Enabled = True
2939 694
2940 695 End Sub
2941 696
2942 697 Private Sub btnDelete_Click()
2943 698 Dim rec As Recordset
2944 699 Dim db1 As Database
2945 700 Dim strSQL As String
2946 701
2947 702 ' Update the drop down list for AMOS
2948 703 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
2949 704 = " & Me.SCourseNumber_PK.Value & " " &
2950 705 "Me.AMOS_PK.RowSource = strSQL
2951 706 Me.ListTargetMOS_FK.RowSource = strSQL
2952 707
2953 708 ' Allows the user to enter an assigned MOS
2954 709 Me!AMOS_PK.Enabled = True
2955 710
2956 711 End Sub
2957 712
2958 713 Private Sub btnDelete_Click()
2959 714 Dim rec As Recordset
2960 715 Dim db1 As Database
2961 716 Dim strSQL As String
2962 717
2963 718 ' Update the drop down list for AMOS
2964 719 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
2965 720 = " & Me.SCourseNumber_PK.Value & " " &
2966 721 "Me.AMOS_PK.RowSource = strSQL
2967 722 Me.ListTargetMOS_FK.RowSource = strSQL
2968 723
2969 724 ' Allows the user to enter an assigned MOS
2970 725 Me!AMOS_PK.Enabled = True
2971 726
2972 727 End Sub
2973 728
2974 729 Private Sub btnDelete_Click()
2975 730 Dim rec As Recordset
2976 731 Dim db1 As Database
2977 732 Dim strSQL As String
2978 733
2979 734 ' Update the drop down list for AMOS
2980 735 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
2981 736 = " & Me.SCourseNumber_PK.Value & " " &
2982 737 "Me.AMOS_PK.RowSource = strSQL
2983 738 Me.ListTargetMOS_FK.RowSource = strSQL
2984 739
2985 740 ' Allows the user to enter an assigned MOS
2986 741 Me!AMOS_PK.Enabled = True
2987 742
2988 743 End Sub
2989 744
2990 745 Private Sub btnDelete_Click()
2991 746 Dim rec As Recordset
2992 747 Dim db1 As Database
2993 748 Dim strSQL As String
2994 749
2995 750 ' Update the drop down list for AMOS
2996 751 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
2997 752 = " & Me.SCourseNumber_PK.Value & " " &
2998 753 "Me.AMOS_PK.RowSource = strSQL
2999 754 Me.ListTargetMOS_FK.RowSource = strSQL
3000 755
3001 756 ' Allows the user to enter an assigned MOS
3002 757 Me!AMOS_PK.Enabled = True
3003 758
3004 759 End Sub
3005 760
3006 761 Private Sub btnDelete_Click()
3007 762 Dim rec As Recordset
3008 763 Dim db1 As Database
3009 764 Dim strSQL As String
3010 765
3011 766 ' Update the drop down list for AMOS
3012 767 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
3013 768 = " & Me.SCourseNumber_PK.Value & " " &
3014 769 "Me.AMOS_PK.RowSource = strSQL
3015 770 Me.ListTargetMOS_FK.RowSource = strSQL
3016 771
3017 772 ' Allows the user to enter an assigned MOS
3018 773 Me!AMOS_PK.Enabled = True
3019 774
3020 775 End Sub
3021 776
3022 777 Private Sub btnDelete_Click()
3023 778 Dim rec As Recordset
3024 779 Dim db1 As Database
3025 780 Dim strSQL As String
3026 781
3027 782 ' Update the drop down list for AMOS
3028 783 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
3029 784 = " & Me.SCourseNumber_PK.Value & " " &
3030 785 "Me.AMOS_PK.RowSource = strSQL
3031 786 Me.ListTargetMOS_FK.RowSource = strSQL
3032 787
3033 788 ' Allows the user to enter an assigned MOS
3034 789 Me!AMOS_PK.Enabled = True
3035 790
3036 791 End Sub
3037 792
3038 793 Private Sub btnDelete_Click()
3039 794 Dim rec As Recordset
3040 795 Dim db1 As Database
3041 796 Dim strSQL As String
3042 797
3043 798 ' Update the drop down list for AMOS
3044 799 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
3045 800 = " & Me.SCourseNumber_PK.Value & " " &
3046 801 "Me.AMOS_PK.RowSource = strSQL
3047 802 Me.ListTargetMOS_FK.RowSource = strSQL
3048 803
3049 804 ' Allows the user to enter an assigned MOS
3050 805 Me!AMOS_PK.Enabled = True
3051 806
3052 807 End Sub
3053 808
3054 809 Private Sub btnDelete_Click()
3055 810 Dim rec As Recordset
3056 811 Dim db1 As Database
3057 812 Dim strSQL As String
3058 813
3059 814 ' Update the drop down list for AMOS
3060 815 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
3061 816 = " & Me.SCourseNumber_PK.Value & " " &
3062 817 "Me.AMOS_PK.RowSource = strSQL
3063 818 Me.ListTargetMOS_FK.RowSource = strSQL
3064 819
3065 820 ' Allows the user to enter an assigned MOS
3066 821 Me!AMOS_PK.Enabled = True
3067 822
3068 823 End Sub
3069 824
3070 825 Private Sub btnDelete_Click()
3071 826 Dim rec As Recordset
3072 827 Dim db1 As Database
3073 828 Dim strSQL As String
3074 829
3075 830 ' Update the drop down list for AMOS
3076 831 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
3077 832 = " & Me.SCourseNumber_PK.Value & " " &
3078 833 "Me.AMOS_PK.RowSource = strSQL
3079 834 Me.ListTargetMOS_FK.RowSource = strSQL
3080 835
3081 836 ' Allows the user to enter an assigned MOS
3082 837 Me!AMOS_PK.Enabled = True
3083 838
3084 839 End Sub
3085 840
3086 841 Private Sub btnDelete_Click()
3087 842 Dim rec As Recordset
3088 843 Dim db1 As Database
3089 844 Dim strSQL As String
3090 845
3091 846 ' Update the drop down list for AMOS
3092 847 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
3093 848 = " & Me.SCourseNumber_PK.Value & " " &
3094 849 "Me.AMOS_PK.RowSource = strSQL
3095 850 Me.ListTargetMOS_FK.RowSource = strSQL
3096 851
3097 852 ' Allows the user to enter an assigned MOS
3098 853 Me!AMOS_PK.Enabled = True
3099 854
3100 855 End Sub
3101 856
3102 857 Private Sub btnDelete_Click()
3103 858 Dim rec As Recordset
3104 859 Dim db1 As Database
3105 860 Dim strSQL As String
3106 861
3107 862 ' Update the drop down list for AMOS
3108 863 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
3109 864 = " & Me.SCourseNumber_PK.Value & " " &
3110 865 "Me.AMOS_PK.RowSource = strSQL
3111 866 Me.ListTargetMOS_FK.RowSource = strSQL
3112 867
3113 868 ' Allows the user to enter an assigned MOS
3114 869 Me!AMOS_PK.Enabled = True
3115 870
3116 871 End Sub
3117 872
3118 873 Private Sub btnDelete_Click()
3119 874 Dim rec As Recordset
3120 875 Dim db1 As Database
3121 876 Dim strSQL As String
3122 877
3123 878 ' Update the drop down list for AMOS
3124 879 strSQL = "SELECT TargetMOS_PK FROM TARGET_MOS WHERE TCourseNumber_PK
3125 880 = " & Me.SCourseNumber_PK
```

```

2600 Application.SetOption "Confirm Action Queries", True
2601
2602 338 DoCmd.DoMenuItem acFormBar, acEditMenu, 8, , acMenuVer70
2603 339 DoCmd.DoMenuItem acFormBar, acEditMenu, 6, , acMenuVer70
2604 340
2605 341 Exit Sub
2606 342
2607 343
2608 344 Err_blnDelete_Click
2609 345 MsgBox Err.Description
2610 346 Resume Exit_blnDelete_Click
2611 347
2612 348 End Sub
2613 349 Private Sub btnClose_Click()
2614 350 On Error GoTo Err_blnClose_Click
2615 351
2616 352
2617 353 DoCmd.Close
2618 354 DoCmd.OpenForm "frmMaintenanceSwitchboard"
2619 355
2620 356 Exit_blnClose_Click:
2621 357 Exit Sub
2622 358
2623 359 Err_blnClose_Click
2624 360 MsgBox Err.Description
2625 361 Resume Exit_blnClose_Click
2626 362
2627 363 End Sub
2628
2629 Form: frmSchoolToPEF_Maint
2630 Code
2631 1 Attribute VB_Name = "Form_frmSchoolToPEF_Maint"
2632 2 Attribute VB_Creatable = True
2633 3 Attribute VB_PredeclaredId = True
2634 4 Attribute VB_Exposed = False
2635 5 Option Compare Database
2636 6 Option Explicit
2637 7
2638 8 Private Sub btnClose_Click()
2639 9 On Error GoTo Err_blnClose_Click
2640 10
2641 11
2642 12 DoCmd.Close
2643 13 DoCmd.OpenForm "frmMaintenanceSwitchboard"
2644 14
2645 15 Exit_blnClose_Click:
2646 16 Exit Sub
2647 17
2648 18 Err_blnClose_Click
2649 19 MsgBox Err.Description
2650 20 Resume Exit_blnClose_Click
2651 21
2652 22
2653 23 End Sub
2654 24
2655 25 Private Sub btnDelete_Click()
2656 26 On Error GoTo Err_blnDelete_Click
2657 27
2658 28
2659 29 DoCmd.DoMenuItem acFormBar, acEditMenu, 8, , acMenuVer70
2660 30 DoCmd.DoMenuItem acFormBar, acEditMenu, 6, , acMenuVer70
2661 31
2662 32 Exit_blnDelete_Click:
2663 33 Exit Sub
2664 34
2665 35 Err_blnDelete_Click:
2666 36 MsgBox Err.Description
2667 37 Resume Exit_blnDelete_Click
2668 38
2669 39
2670 40 End Sub
2671 41 Private Sub cmbCourseNumberFind_AfterUpdate()
2672 42
2673 43 Dim R As Recordset
2674 44 Set R = Me.RecordsetClone
2675 45 R.FindFirst "[SCourseNumber_PK] = " & Chr(34) &
2676 46 Me([cmbCourseNumberFind]) & Chr(34)
2677 47 Me.Bookmark = R.Bookmark
2678 48 Me([cmbCourseNumber_Find]) = Null
2679 49 Me.SelCourseNumber_PK.SetFocus
2680 50
2681 51 End Sub
2682 52 Private Sub Form_Current()
2683 53 Dim strSQL As String
2684 54
2685 55 ' Update the Potential PEF list
2686 56 strSQL = "SELECT DISTINCTROW [PEF].[PEF_PK] FROM PEF LEFT JOIN
2687 57 qrySch_Pef ON ([PEF].[PEF_PK] = [qrySch_Pef].[PEF_PK] WHERE
2688 58 ([qrySch_Pef].[PEF_PK] Is Null) ORDER BY [PEF].[PEF_PK];"
2689 59
2690 60 Me.lstUnselectedPEF.RowSource = strSQL
2691 61
2692 62 ' Update the Selected PEF List
2693 63 strSQL = "SELECT [PEF_PK] FROM qrySch_Pef;"
2694 64 Me.lstSelectedPEF.RowSource = strSQL
2695 65
2696 66 End Sub
2697 67 Private Sub btnLeft_Click()
2698 68 Dim frm As Form, ctl As Control
2699 69 Dim varItem As Variant, int As Integer
2700 70 Dim strSQL As String
2701 71
2702 72 ' Ensures the current record is save to the SCH_LOG table
2703 73 DoCmd.DoMenuItem acFormBar, acRecordsMenu, acSaveRecord, ,
2704 74
2705 75 ' This code enters the selected Target MOS's into the SCH_TGT_MOS
2706 76 Set cti = frm.lstUnselectedPEF
2707 77 For Each varItem In cti.ItemsSelected
2708 78 For int = 0 To cti.ColumnCount - 1
2709 79
2710 80 ' This puts the value found in the list box into a text box,
2711 81 making it readable by the query
2712 82 Form!frmSchoolToPEF_Maint!PEFValue.Value = cti.Column(int),
2713 83
2714 84 ' Disables the action query confirmation message
2715 85 Application.SetOption "Confirm Action Queries", False
2716 86
2717 87 ' Enters the value into the SCH_TGT_MOS table
2718 88 DoCmd.OpenQuery "qryUpdateSCH_PEF"
2719 89
2720 90 ' Enables the action query confirmation message
2721 91 Application.SetOption "Confirm Action Queries", True
2722 92
2723 93 Next int
2724 94 Next varItem
2725 95
2726 96 ' Update the Potential PEF list
2727 97 strSQL = "SELECT DISTINCTROW [PEF].[PEF_PK] FROM PEF LEFT JOIN
2728 98

```

```

2730 qrySch_Pef ON ([PEF].[PEF_PK] = [qrySch_Pef].[PEF_PK] WHERE
2731 ([qrySch_Pef].[PEF_PK] Is Null) ORDER BY [PEF].[PEF_PK];"
2732 99 Me.lstUnselectedPEF.RowSource = strSQL
2733 100
2734 101 ' Update the Selected PEF List
2735 102 strSQL = "SELECT [PEF_PK] FROM qrySch_Pef;"
2736 103 Me.lstSelectedPEF.RowSource = strSQL
2737 104
2738 105 End Sub
2739 106
2740 107 Private Sub btnRight_Click()
2741 108 Dim frm As Form, ctl As Control
2742 109 Dim varItem As Variant, int As Integer
2743 110 Dim strSQL As String
2744 111
2745 112 ' This code Deletes the selected Target MOS's and its associated
2746 113 entries from the SCH_TGT_MOS table
2747 114 Set frm = Form!frmSchoolToPEF_Maint
2748 115 Set cti = frm.lstSelectedPEF
2749 116 For Each varItem In cti.ItemsSelected
2750 117 For int = 0 To cti.ColumnCount - 1
2751 118
2752 119 ' This puts the value found in the list box into a text box,
2753 120 making it readable by the query
2754 121 Form!frmSchoolToPEF_Maint!PEFValue.Value = cti.Column(int),
2755 122
2756 123 ' Disables the action query confirmation message
2757 124 Application.SetOption "Confirm Action Queries", False
2758 125
2759 126 ' Deletes the record associated with the specified value from
2760 127 the SCH_TGT_MOS table
2761 128 DoCmd.OpenQuery "qryDeleteSCH_PEF"
2762 129
2763 130 ' Enables the action query confirmation message
2764 131 Application.SetOption "Confirm Action Queries", True
2765 132
2766 133 Next int
2767 134 Next varItem
2768 135
2769 136 ' Update the Potential PEF list
2770 137 strSQL = "SELECT DISTINCTROW [PEF].[PEF_PK] FROM PEF LEFT JOIN
2771 138 qrySch_Pef ON ([PEF].[PEF_PK] = [qrySch_Pef].[PEF_PK] WHERE
2772 139 ([qrySch_Pef].[PEF_PK] Is Null) ORDER BY [PEF].[PEF_PK];"
2773 140 Me.lstUnselectedPEF.RowSource = strSQL
2774 141
2775 142 ' Update the Selected PEF List
2776 143 strSQL = "SELECT [PEF_PK] FROM qrySch_Pef;"
2777 144 Me.lstSelectedPEF.RowSource = strSQL
2778 145
2779 146 End Sub
2780
2781 Form: frmSchoolToPropertyMaint
2782 Code
2783 1 Attribute VB_Name = "Form_frmSchoolToPropertyMaint"
2784 2 Attribute VB_Creatable = True
2785 3 Attribute VB_PredeclaredId = True
2786 4 Attribute VB_Exposed = False
2787 5 Option Compare Database
2788 6 Option Explicit
2789 7
2790 8 Private Sub btnClose_Click()
2791 9 On Error GoTo Err_blnClose_Click
2792 10
2793 11
2794 12 DoCmd.Close
2795 13 DoCmd.OpenForm "frmMaintenanceSwitchboard"
2796 14
2797 15 Exit_blnClose_Click:
2798 16 Exit Sub
2799 17
2800 18 Err_blnClose_Click
2801 19 MsgBox Err.Description
2802 20 Resume Exit_blnClose_Click
2803 21
2804 22 End Sub
2805 23 Private Sub btnDelete_Click()
2806 24 On Error GoTo Err_blnDelete_Click
2807 25
2808 26
2809 27 DoCmd.DoMenuItem acFormBar, acEditMenu, 8, , acMenuVer70
2810 28 DoCmd.DoMenuItem acFormBar, acEditMenu, 6, , acMenuVer70
2811 29
2812 30 Exit_blnDelete_Click:
2813 31 Exit Sub
2814 32
2815 33 Err_blnDelete_Click:
2816 34 MsgBox Err.Description
2817 35 Resume Exit_blnDelete_Click
2818 36
2819 37 End Sub
2820 38 Private Sub btnLevel_Click()
2821 39 On Error GoTo Err_blnLevel_Click
2822 40
2823 41 Dim lngRecordNum As Long
2824 42 Dim strSQL As String
2825 43
2826 44 lngRecordNum = Me.CurrentRecord
2827 45
2828 46 DoCmd.Close
2829 47 DoCmd.OpenForm "frmSchoolToPropLevelMaint"
2830 48 DoCmd.OpenForm "frmSchoolToPropLevelMaint", acGoTo,
2831 49 lngRecordNum
2832 50 Form!frmSchoolToPropLevelMaint!ChosenLevel = 0
2833 51
2834 52 ' Update the selected property list
2835 53 strSQL = "SELECT [Property Name_PK] FROM FUND_SCH_PROP WHERE
2836 54 SCourseNumber_PK = " &
2837 55 Form!frmSchoolToPropLevelMaint!SCourseNumber_PK.Value & "" & " And
2838 56 AMOS_FK = " & Form!frmSchoolToPropLevelMaint!AMOS_PK.Value & "" & "
2839 57 And Level = " & Form!frmSchoolToPropLevelMaint!ChosenLevel & "" & "
2840 58 UNION SELECT ([Property Name_PK] FROM LOG_SCH_PROP WHERE SCourseNumber_PK
2841 59 = " & Form!frmSchoolToPropLevelMaint!SCourseNumber_PK.Value & "" & "
2842 60 And AMOS_FK = " & Form!frmSchoolToPropLevelMaint!AMOS_PK.Value & "" & "
2843 61 And Level = " & Form!frmSchoolToPropLevelMaint!ChosenLevel & "" & "
2844 62 Form!frmSchoolToPropLevelMaint!listLevel.RowSource = strSQL
2845 63
2846 64 Exit_blnLevel_Click:
2847 65 Exit Sub
2848 66 Err_blnLevel_Click:
2849 67 MsgBox Err.Description
2850 68 Resume Exit_blnLevel_Click
2851 69
2852 70 End Sub
2853 71
2854 72
2855 73
2856 74
2857 75
2858 76
2859 77

```



```

2860 69 Private Sub btnLevel1_Click()
2861 70 On Error GoTo Err_btnLevel1_Click
2862 71
2863 72 Dim lngRecordNum As Long
2864 73 Dim strSQL As String
2865 74
2866 75 lngRecordNum = Me.CurrentRecord
2867 76
2868 77 DoCmd.Close
2869 78
2870 79 DoCmd.OpenForm "frmSchoolToPropLevelMaint"
2871 80 DoCmd.GoToRecord acDataForm, "frmSchoolToPropLevelMaint", acGoTo,
2872 lngRecordNum
2873 81 Form!frmSchoolToPropLevelMaint!ChosenLevel = 1
2874 82
2875 83 ' Update the selected property list
2876 84 strSQL = "SELECT (F!PropertyName_RK) FROM FUND_SCH_PROP WHERE
2877 S!CourseNumber_RK = " &
2878 Form!frmSchoolToPropLevelMaint!S!CourseNumber_RK.Value & " " & " And
2879 AMOS_RK = " & Form!frmSchoolToPropLevelMaint!AMOS_RK.Value & " " & "
2880 And Level = " & Form!frmSchoolToPropLevelMaint!ChosenLevel & " " & "
2881 UNION SELECT (L!PropertyName_RK) FROM LOG_SCH_PROP WHERE S!CourseNumber_RK
2882 = " & Form!frmSchoolToPropLevelMaint!S!CourseNumber_RK.Value & " " & "
2883 And AMOS_RK = " & Form!frmSchoolToPropLevelMaint!AMOS_RK.Value & " " & "
2884 And Level = " & Form!frmSchoolToPropLevelMaint!ChosenLevel & " " & "
2885 Form!frmSchoolToPropLevelMaint!Level0.RowSource = strSQL
2886 85
2887 86 Exit btnLevel1_Click
2888 87
2889 88 Err_btnLevel1_Click
2890 89
2891 90 MsgBox Err.Description
2892 91 Resume Exit_btnLevel1_Click
2893 92
2894 93 End Sub
2895 94
2896 95 Private Sub btnLevel2_Click()
2897 96 On Error GoTo Err_btnLevel2_Click
2898 97
2899 98 Dim lngRecordNum As Long
2900 99 Dim strSQL As String
2901 100
2902 101 lngRecordNum = Me.CurrentRecord
2903 102
2904 103 DoCmd.Close
2905 104
2906 105 DoCmd.OpenForm "frmSchoolToPropLevelMaint"
2907 106 DoCmd.GoToRecord acDataForm, "frmSchoolToPropLevelMaint", acGoTo,
2908 lngRecordNum
2909 107 Form!frmSchoolToPropLevelMaint!ChosenLevel = 2
2910 108
2911 109 ' Update the selected property list
2912 110 strSQL = "SELECT (F!PropertyName_RK) FROM FUND_SCH_PROP WHERE
2913 S!CourseNumber_RK = " &
2914 Form!frmSchoolToPropLevelMaint!S!CourseNumber_RK.Value & " " & " And
2915 AMOS_RK = " & Form!frmSchoolToPropLevelMaint!AMOS_RK.Value & " " & "
2916 And Level = " & Form!frmSchoolToPropLevelMaint!ChosenLevel & " " & "
2917 UNION SELECT (L!PropertyName_RK) FROM LOG_SCH_PROP WHERE S!CourseNumber_RK
2918 = " & Form!frmSchoolToPropLevelMaint!S!CourseNumber_RK.Value & " " & "
2919 And AMOS_RK = " & Form!frmSchoolToPropLevelMaint!AMOS_RK.Value & " " & "
2920 And Level = " & Form!frmSchoolToPropLevelMaint!ChosenLevel & " " & "
2921 Form!frmSchoolToPropLevelMaint!Level0.RowSource = strSQL
2922 111
2923 112 Exit btnLevel2_Click
2924 113
2925 114 Err_btnLevel2_Click
2926 115
2927 116 MsgBox Err.Description
2928 117 Resume Exit_btnLevel2_Click
2929 118
2930 119 End Sub
2931 120
2932 121 Private Sub btnLevel3_Click()
2933 122 On Error GoTo Err_btnLevel3_Click
2934 123
2935 124 Dim lngRecordNum As Long
2936 125 Dim strSQL As String
2937 126
2938 127 lngRecordNum = Me.CurrentRecord
2939 128
2940 129 DoCmd.Close
2941 130
2942 131 DoCmd.OpenForm "frmSchoolToPropLevelMaint"
2943 132 DoCmd.GoToRecord acDataForm, "frmSchoolToPropLevelMaint", acGoTo,
2944 lngRecordNum
2945 133 Form!frmSchoolToPropLevelMaint!ChosenLevel = 3
2946 134
2947 135 ' Update the selected property list
2948 136 strSQL = "SELECT (F!PropertyName_RK) FROM FUND_SCH_PROP WHERE
2949 S!CourseNumber_RK = " &
2950 Form!frmSchoolToPropLevelMaint!S!CourseNumber_RK.Value & " " & " And
2951 AMOS_RK = " & Form!frmSchoolToPropLevelMaint!AMOS_RK.Value & " " & "
2952 And Level = " & Form!frmSchoolToPropLevelMaint!ChosenLevel & " " & "
2953 UNION SELECT (L!PropertyName_RK) FROM LOG_SCH_PROP WHERE S!CourseNumber_RK
2954 = " & Form!frmSchoolToPropLevelMaint!S!CourseNumber_RK.Value & " " & "
2955 And AMOS_RK = " & Form!frmSchoolToPropLevelMaint!AMOS_RK.Value & " " & "
2956 And Level = " & Form!frmSchoolToPropLevelMaint!ChosenLevel & " " & "
2957 Form!frmSchoolToPropLevelMaint!Level0.RowSource = strSQL
2958 137
2959 138 Exit btnLevel3_Click
2960 139
2961 140 Err_btnLevel3_Click
2962 141
2963 142 MsgBox Err.Description
2964 143 Resume Exit_btnLevel3_Click
2965 144
2966 145 End Sub
2967 146
2968 147 Private Sub btnLevel4_Click()
2969 148 On Error GoTo Err_btnLevel4_Click
2970 149
2971 150 Dim lngRecordNum As Long
2972 151 Dim strSQL As String
2973 152
2974 153 lngRecordNum = Me.CurrentRecord
2975 154
2976 155 DoCmd.Close
2977 156
2978 157 DoCmd.OpenForm "frmSchoolToPropLevelMaint"
2979 158 DoCmd.GoToRecord acDataForm, "frmSchoolToPropLevelMaint", acGoTo,
2980 lngRecordNum
2981 159 Form!frmSchoolToPropLevelMaint!ChosenLevel = 4
2982 160
2983 161 ' Update the selected property list
2984 162 strSQL = "SELECT (F!PropertyName_RK) FROM FUND_SCH_PROP WHERE
2985 S!CourseNumber_RK = " &
2986 Form!frmSchoolToPropLevelMaint!S!CourseNumber_RK.Value & " " & " And
2987 AMOS_RK = " & Form!frmSchoolToPropLevelMaint!AMOS_RK.Value & " " & "
2988 And Level = " & Form!frmSchoolToPropLevelMaint!ChosenLevel & " " & "
2989 UNION SELECT (L!PropertyName_RK) FROM LOG_SCH_PROP WHERE S!CourseNumber_RK

```

```

2990 = " & Form!frmSchoolToPropLevelMaint!S!CourseNumber_RK.Value & " " & "
2991 And AMOS_RK = " & Form!frmSchoolToPropLevelMaint!AMOS_RK.Value & " " & "
2992 And Level = " & Form!frmSchoolToPropLevelMaint!ChosenLevel & " " & "
2993 Form!frmSchoolToPropLevelMaint!Level0.RowSource = strSQL
2994 163
2995 164 Exit btnLevel4_Click
2996 165
2997 166 Err_btnLevel4_Click
2998 167
2999 168 MsgBox Err.Description
3000 169 Resume Exit_btnLevel4_Click
3001 170
3002 171 End Sub
3003 172
3004 173 Private Sub btnLevel5_Click()
3005 174 On Error GoTo Err_btnLevel5_Click
3006 175
3007 176 Dim lngRecordNum As Long
3008 177 Dim strSQL As String
3009 178
3010 179 lngRecordNum = Me.CurrentRecord
3011 180
3012 181 DoCmd.Close
3013 182
3014 183 DoCmd.OpenForm "frmSchoolToPropLevelMaint"
3015 184 DoCmd.GoToRecord acDataForm, "frmSchoolToPropLevelMaint", acGoTo,
3016 lngRecordNum
3017 185 Form!frmSchoolToPropLevelMaint!ChosenLevel = 5
3018 186
3019 187 ' Update the selected property list
3020 188 strSQL = "SELECT (F!PropertyName_RK) FROM FUND_SCH_PROP WHERE
3021 S!CourseNumber_RK = " &
3022 Form!frmSchoolToPropLevelMaint!S!CourseNumber_RK.Value & " " & " And
3023 AMOS_RK = " & Form!frmSchoolToPropLevelMaint!AMOS_RK.Value & " " & "
3024 And Level = " & Form!frmSchoolToPropLevelMaint!ChosenLevel & " " & "
3025 UNION SELECT (L!PropertyName_RK) FROM LOG_SCH_PROP WHERE S!CourseNumber_RK
3026 = " & Form!frmSchoolToPropLevelMaint!S!CourseNumber_RK.Value & " " & "
3027 And AMOS_RK = " & Form!frmSchoolToPropLevelMaint!AMOS_RK.Value & " " & "
3028 And Level = " & Form!frmSchoolToPropLevelMaint!ChosenLevel & " " & "
3029 Form!frmSchoolToPropLevelMaint!Level0.RowSource = strSQL
3030 189
3031 190 Exit btnLevel5_Click
3032 191
3033 192 Err_btnLevel5_Click
3034 193
3035 194 MsgBox Err.Description
3036 195 Resume Exit_btnLevel5_Click
3037 196
3038 197 End Sub
3039 198
3040 199 Private Sub btnLevel6_Click()
3041 200 On Error GoTo Err_btnLevel6_Click
3042 201
3043 202 Dim lngRecordNum As Long
3044 203 Dim strSQL As String
3045 204
3046 205 lngRecordNum = Me.CurrentRecord
3047 206
3048 207 DoCmd.Close
3049 208
3050 209 DoCmd.OpenForm "frmSchoolToPropLevelMaint"
3051 210 DoCmd.GoToRecord acDataForm, "frmSchoolToPropLevelMaint", acGoTo,
3052 lngRecordNum
3053 211 Form!frmSchoolToPropLevelMaint!ChosenLevel = 6
3054 212
3055 213 ' Update the selected property list
3056 214 strSQL = "SELECT (F!PropertyName_RK) FROM FUND_SCH_PROP WHERE
3057 S!CourseNumber_RK = " &
3058 Form!frmSchoolToPropLevelMaint!S!CourseNumber_RK.Value & " " & " And
3059 AMOS_RK = " & Form!frmSchoolToPropLevelMaint!AMOS_RK.Value & " " & "
3060 And Level = " & Form!frmSchoolToPropLevelMaint!ChosenLevel & " " & "
3061 UNION SELECT (L!PropertyName_RK) FROM LOG_SCH_PROP WHERE S!CourseNumber_RK
3062 = " & Form!frmSchoolToPropLevelMaint!S!CourseNumber_RK.Value & " " & "
3063 And AMOS_RK = " & Form!frmSchoolToPropLevelMaint!AMOS_RK.Value & " " & "
3064 And Level = " & Form!frmSchoolToPropLevelMaint!ChosenLevel & " " & "
3065 Form!frmSchoolToPropLevelMaint!Level0.RowSource = strSQL
3066 215
3067 216 Exit btnLevel6_Click
3068 217
3069 218 Err_btnLevel6_Click
3070 219
3071 220 MsgBox Err.Description
3072 221 Resume Exit_btnLevel6_Click
3073 222
3074 223 End Sub
3075 224
3076 225 Private Sub cmbCourseNumberFind_AfterUpdate()
3077 226 Dim R As Recordset
3078 227 Set R = Me.RecordsetClone
3079 228 R.FindFirst "[S!CourseNumber_RK] = " & Chr(34) &
3080 Me!cmbCourseNumberFind & Chr(34)
3081 229
3082 230 Me!cmbCourseNumberFind = Null
3083 231 Me!S!CourseNumber_RK.SetFocus
3084 232
3085 233 End Sub
3086 234
3087 235 Private Sub Form_Current()
3088 236 Dim strSQL As String
3089 237 Dim lngLevel As Integer
3090 238
3091 239 ' Update the Level 0 list
3092 240 lngLevel = 0
3093 241 strSQL = "SELECT (F!PropertyName_RK) FROM FUND_SCH_PROP WHERE
3094 S!CourseNumber_RK = " & Me!S!CourseNumber_RK.Value & " " & " And AMOS_RK =
3095 " & Me!AMOS_RK.Value & " " & " And Level = " & lngLevel & " " & " UNION
3096 SELECT (L!PropertyName_RK) FROM LOG_SCH_PROP WHERE S!CourseNumber_RK = " &
3097 Me!S!CourseNumber_RK.Value & " " & " And AMOS_RK = " & Me!AMOS_RK.Value
3098 & " " & " And Level = " & lngLevel & " " & "
3099 Me!Level0.RowSource = strSQL
3100 242
3101 243 ' Update the Level 1 list
3102 244 lngLevel = 1
3103 245 strSQL = "SELECT (F!PropertyName_RK) FROM FUND_SCH_PROP WHERE
3104 S!CourseNumber_RK = " & Me!S!CourseNumber_RK.Value & " " & " And AMOS_RK =
3105 " & Me!AMOS_RK.Value & " " & " And Level = " & lngLevel & " " & " UNION
3106 SELECT (L!PropertyName_RK) FROM LOG_SCH_PROP WHERE S!CourseNumber_RK = " &
3107 Me!S!CourseNumber_RK.Value & " " & " And AMOS_RK = " & Me!AMOS_RK.Value
3108 & " " & " And Level = " & lngLevel & " " & "
3109 Me!Level1.RowSource = strSQL
3110 246
3111 247 ' Update the Level 2 list
3112 248 lngLevel = 2
3113 249 strSQL = "SELECT (F!PropertyName_RK) FROM FUND_SCH_PROP WHERE
3114 S!CourseNumber_RK = " & Me!S!CourseNumber_RK.Value & " " & " And AMOS_RK =
3115 " & Me!AMOS_RK.Value & " " & " And Level = " & lngLevel & " " & " UNION
3116 SELECT (L!PropertyName_RK) FROM LOG_SCH_PROP WHERE S!CourseNumber_RK = " &
3117 Me!S!CourseNumber_RK.Value & " " & " And AMOS_RK = " & Me!AMOS_RK.Value
3118 & " " & " And Level = " & lngLevel & " " & "
3119 Me!Level2.RowSource = strSQL

```

```

3120 & "" & " And Level = " & intLevel & ""
3121 Me.lstLevel0.RowSource = strSQL
3122
3123
3124 261
3125
3126 262
3127
3128 263
3129
3130 264
3131
3132 265
3133
3134 266
3135
3136 267
3137
3138 268
3139
3140 269
3141
3142 270
3143
3144 271
3145
3146 272
3147
3148 273
3149
3150 274
3151
3152 275
3153
3154 276
3155
3156 277
3157
3158 278
3159
3160 279
3161
3162 Form: frmSchoolToPropLevelMaint
3163 Code
3164 1 Attribute VB_Name = "Form_frmSchoolToPropLevelMaint"
3165 2 Attribute VB_Creatable = True
3166 3 Attribute VB_PredeclaredId = True
3167 4 Attribute VB_Exposed = False
3168 5 Option Compare Database
3169 6 Option Explicit
3170 7
3171 8 Private Sub btnClose_Click()
3172 9 On Error GoTo Err_btnClose_Click
3173 10
3174 11 Dim lngRecordNum As Long
3175 12
3176 13 lngRecordNum = Me.CurrentRecord
3177 14
3178 15 DoCmd.Close
3179 16
3180 17 DoCmd.OpenForm "frmSchoolToPropertyMaint"
3181 18 DoCmd.GoToRecord acDataForm, "frmSchoolToPropertyMaint", acGoTo,
3182 lngRecordNum
3183 19
3184 20 Exit_btnClose_Click
3185 21 Exit Sub
3186 22
3187 23 Err_btnClose_Click
3188 24 MsgBox Err.Description
3189 25 Resume Err_btnClose_Click
3190 26
3191 27 End Sub
3192 28
3193 29 Private Sub btnDelete_Click()
3194 30 On Error GoTo Err_btnDelete_Click
3195 31
3196 32
3197 33 DoCmd.DoMenuItem acFormBar, acEditMenu, 8, , acMenuVer70
3198 34 DoCmd.DoMenuItem acFormBar, acEditMenu, 6, , acMenuVer70
3199 35
3200 36 Exit_btnDelete_Click
3201 37 Exit Sub
3202 38
3203 39 Err_btnDelete_Click
3204 40 MsgBox Err.Description
3205 41 Resume Err_btnDelete_Click
3206 42
3207 43 End Sub
3208 44
3209 45 Private Sub ChosenLevel_AfterUpdate()
3210 46
3211 47 Dim strSQL As String
3212 48
3213 49 ' Update the potential property list
3214 50 strSQL = "SELECT DISTINCTROW [qryUnionAllProperty].[FPropertyName_FK]
3215 FROM qryUnionAllProperty LEFT JOIN qryUnionPropertyByLevel ON
3216 [qryUnionPropertyByLevel].[FPropertyName_FK] =
3217 [qryUnionPropertyByLevel].[FPropertyName_FK] WHERE
3218 ([qryUnionPropertyByLevel].[FPropertyName_FK] Is Null);"
3219 51 Me.lstLevelUnUsed.RowSource = strSQL
3220 52
3221 53 strSQL = "SELECT [FPropertyName_FK] FROM FUND_SCH_PROP WHERE
3222 CourseNumber_FK = " & Me.CourseNumber_PK.Value & "" & " And AMOS_FK =
3223 " & Me.AMOS_PK.Value & "" & " And Level = " & Me.ChosenLevel.Value &
3224 " & " UNION SELECT [LPropertyName_FK] FROM LOG_SCH_PROP WHERE
3225 CourseNumber_FK = " & Me.CourseNumber_PK.Value & "" & " And AMOS_FK =
3226 " & Me.AMOS_PK.Value & "" & " And Level = " & Me.ChosenLevel.Value &
3227
3228 54 Me.lstLevel0.RowSource = strSQL
3229 55
3230 56 End Sub
3231 57
3232 58 Private Sub cmbCourseNumberFind_AfterUpdate()
3233 59
3234 60 Dim R As Recordset
3235 61 Set R = Me.RecordsetClone
3236 62 R.FindFirst "CourseNumber_PK = " & Chr(34) &
3237 Me[cmbCourseNumberFind] & Chr(34)
3238 63 Me.Bookmark = R.Bookmark
3239 64 Me[cmbCourseNumberFind] = Null
3240 65 Me.CourseNumber_PK.SetFocus
3241 66
3242 67 End Sub
3243 68
3244 69 Private Sub Form_Current()
3245 70
3246 71 Dim strSQL As String
3247 72
3248 73 ' Update the potential property list
3249 74 strSQL = "SELECT DISTINCTROW [qryUnionAllProperty].[FPropertyName_FK]

```

```

3250 FROM qryUnionAllProperty LEFT JOIN qryUnionPropertyByLevel ON
3251 [qryUnionPropertyByLevel].[FPropertyName_FK] =
3252 [qryUnionPropertyByLevel].[FPropertyName_FK] WHERE
3253 ([qryUnionPropertyByLevel].[FPropertyName_FK] Is Null);"
3254 75 Me.lstLevelUnUsed.RowSource = strSQL
3255 76
3256 77 ' Update the selected property list
3257 78 strSQL = "SELECT [FPropertyName_FK] FROM FUND_SCH_PROP WHERE
3258 CourseNumber_FK = " & Me.CourseNumber_PK.Value & "" & " And AMOS_FK =
3259 " & Me.AMOS_PK.Value & "" & " And Level = " & Me.ChosenLevel.Value &
3260 " & " UNION SELECT [LPropertyName_FK] FROM LOG_SCH_PROP WHERE
3261 CourseNumber_FK = " & Me.CourseNumber_PK.Value & "" & " And AMOS_FK =
3262 " & Me.AMOS_PK.Value & "" & " And Level = " & Me.ChosenLevel.Value &
3263
3264 79 Me.lstLevel0.RowSource = strSQL
3265 80
3266 81 End Sub
3267 82
3268 83 Private Sub LeftArrow_Click()
3269 84 Dim frm As Form, ctl As Control
3270 85 Dim varItem As Variant, int As Integer
3271 86 Dim strSQL As String
3272 87
3273 88 ' Ensures the current record is save to the SCHOOL table
3274 89 DoCmd.DoMenuItem acFormBar, acRecordsMenu, acSaveRecord, ,
3275
3276 90 ' This code enters the selected Target MOS's into the SCH_TGT_MOS
3277 91 Set frm = Form!frmSchoolToPropLevelMaint
3278 92 Set ctl = frm!lstLevelUnUsed
3279 93 For Each varItem In ctl.ItemsSelected
3280 94 For int = 0 To ctl.ColumnCount - 1
3281 95
3282 96 ' This puts the value found in the list box into a text box
3283 97 making it readable by the query
3284 98 Format!frmSchoolToPropLevelMaint!TargetValue.Value =
3285 ctl.Column(int, varItem)
3286 99
3287 100 ' Disables the action query confirmation message
3288 101 Application.SetOption "Confirm Action Queries", False
3289 102 DoCmd.SetWarnings (False)
3290 103
3291 104 ' Enters the value into the SCH_TGT_MOS table
3292 105 DoCmd.OpenQuery "qryUpdateFUND_SCH_PROP"
3293 106 DoCmd.OpenQuery "qryUpdateLOG_SCH_PROP"
3294 107
3295 108 ' Enables the action query confirmation message
3296 109 Application.SetOption "Confirm Action Queries", True
3297 110 DoCmd.SetWarnings (True)
3298 111
3299 112 Next int
3300 113 Next varItem
3301 114
3302 115 ' Update the potential property list
3303 116 strSQL = "SELECT DISTINCTROW [qryUnionAllProperty].[FPropertyName_FK]
3304 FROM qryUnionAllProperty LEFT JOIN qryUnionPropertyByLevel ON
3305 [qryUnionPropertyByLevel].[FPropertyName_FK] =
3306 [qryUnionPropertyByLevel].[FPropertyName_FK] WHERE
3307 ([qryUnionPropertyByLevel].[FPropertyName_FK] Is Null);"
3308 117 Me.lstLevelUnUsed.RowSource = strSQL
3309 118
3310 119 ' Update the selected property list
3311 120 strSQL = "SELECT [FPropertyName_FK] FROM FUND_SCH_PROP WHERE
3312 CourseNumber_FK = " & Me.CourseNumber_PK.Value & "" & " And AMOS_FK =
3313 " & Me.AMOS_PK.Value & "" & " And Level = " & Me.ChosenLevel.Value &
3314 " & " UNION SELECT [LPropertyName_FK] FROM LOG_SCH_PROP WHERE
3315 CourseNumber_FK = " & Me.CourseNumber_PK.Value & "" & " And AMOS_FK =
3316 " & Me.AMOS_PK.Value & "" & " And Level = " & Me.ChosenLevel.Value &
3317
3318 121 Me.lstLevel0.RowSource = strSQL
3319 122 End Sub
3320 123
3321 124 Private Sub RightArrow_Click()
3322 125 Dim frm As Form, ctl As Control
3323 126 Dim varItem As Variant, int As Integer
3324 127 Dim strSQL As String
3325 128
3326 129 ' This code Deletes the selected Target MOS's and its associated
3327 130 entries from the SCH_TGT_MOS table
3328 131 Set frm = Form!frmSchoolToPropLevelMaint
3329 132 Set ctl = frm!lstLevelUnUsed
3330 133 For Each varItem In ctl.ItemsSelected
3331 134 For int = 0 To ctl.ColumnCount - 1
3332 135
3333 136 ' This puts the value found in the list box into a text box
3334 137 making it readable by the query
3335 138 Format!frmSchoolToPropLevelMaint!TargetValue.Value =
3336 ctl.Column(int, varItem)
3337 139
3338 140 ' Disables the action query confirmation message
3339 141 Application.SetOption "Confirm Action Queries", False
3340 142 DoCmd.SetWarnings (False)
3341 143
3342 144 ' Deletes the record associated with the specified value from
3343 145 the FUND or LOG_SCH_PROP table
3344 146 DoCmd.OpenQuery "qryDeleteFUND_SCH_PROP"
3345 147 DoCmd.OpenQuery "qryDeleteLOG_SCH_PROP"
3346 148
3347 149 ' Enables the action query confirmation message
3348 150 Application.SetOption "Confirm Action Queries", True
3349 151 DoCmd.SetWarnings (True)
3350 152
3351 153 Next int
3352 154 Next varItem
3353 155
3354 156 ' Update the potential property list
3355 157 strSQL = "SELECT DISTINCTROW [qryUnionAllProperty].[FPropertyName_FK]
3356 FROM qryUnionAllProperty LEFT JOIN qryUnionPropertyByLevel ON
3357 [qryUnionPropertyByLevel].[FPropertyName_FK] =
3358 [qryUnionPropertyByLevel].[FPropertyName_FK] WHERE
3359 ([qryUnionPropertyByLevel].[FPropertyName_FK] Is Null);"
3360 158 Me.lstLevelUnUsed.RowSource = strSQL
3361 159
3362 160 ' Update the selected property list
3363 161 strSQL = "SELECT [FPropertyName_FK] FROM FUND_SCH_PROP WHERE
3364 CourseNumber_FK = " & Me.CourseNumber_PK.Value & "" & " And AMOS_FK =
3365 " & Me.AMOS_PK.Value & "" & " And Level = " & Me.ChosenLevel.Value &
3366 " & " UNION SELECT [LPropertyName_FK] FROM LOG_SCH_PROP WHERE
3367 CourseNumber_FK = " & Me.CourseNumber_PK.Value & "" & " And AMOS_FK =
3368 " & Me.AMOS_PK.Value & "" & " And Level = " & Me.ChosenLevel.Value &
3369
3370 162 Me.lstLevel0.RowSource = strSQL
3371 163
3372 164 End Sub
3373 165
3374 Form: frmScrubMarine
3375 Code
3376 1 Attribute VB_Name = "Form_frmScrubMarine"
3377 2 Attribute VB_Creatable = True
3378 3 Attribute VB_PredeclaredId = True
3379 4 Attribute VB_Exposed = False
3380 5 Option Compare Database

```

```

3380 6 Option Explicit
3381 7
3382 8 Private Sub Close_Click()
3383 9 On Error GoTo Err_Close_Click
3384 10
3385 11 Dim strDocName As String
3386 12 Dim strLinkCriteria As String
3387 13
3388 14 ' Close current form
3389 15 DoCmd.Close
3390 16
3391 17 ' Open specified form
3392 18 strDocName = "frmPreprocessing&ExecutionSwitchboard"
3393 19 DoCmd.OpenForm strDocName, , , strLinkCriteria
3394 20
3395 21 Exit_Close_Click:
3396 22 Exit Sub
3397 23
3398 24 Err_Close_Click:
3399 25 MsgBox Err.Description
3400 26 Resume Exit_Close_Click
3401 27
3402 28 End Sub
3403 29
3404 30 Private Sub cmbPefFind_AfterUpdate()
3405 31
3406 32 Dim R As Recordset
3407 33 Set R = Me.RecordsetClone
3408 34 R.FindFirst "[PEF] = " & Chr(34) & Me[cmbPefFind] & Chr(34)
3409 35 Me.Bookmark = R.Bookmark
3410 36 Me[cmbPefFind] = Null
3411 37 Me.SSN_PK.SetFocus
3412 38
3413 39 End Sub
3414 40
3415 41 Private Sub cmbSSNFind_AfterUpdate()
3416 42
3417 43 Dim R As Recordset
3418 44 Set R = Me.RecordsetClone
3419 45 R.FindFirst "[SSN_PK] = " & Chr(34) & Me[cmbSSNFind] & Chr(34)
3420 46 Me.Bookmark = R.Bookmark
3421 47 Me[cmbSSNFind] = Null
3422 48 Me.Close.SetFocus
3423 49
3424 50 End Sub
3425 51
3426 52 Private Sub Delete_Click()
3427 53 On Error GoTo Err_Delete_Click
3428 54
3429 55 DoCmd.DoMenuItem acFormBar, acEditMenu, 8, , acMenuVer70
3430 56 DoCmd.DoMenuItem acFormBar, acEditMenu, 6, , acMenuVer70
3431 57
3432 58 Exit_Delete_Click:
3433 59 Exit Sub
3434 60
3435 61 Err_Delete_Click:
3436 62 MsgBox Err.Description
3437 63 Resume Exit_Delete_Click
3438 64
3439 65 End Sub
3440 66
3441 67 Private Sub Form_Current()
3442 68 Dim strSQL As String
3443 69
3444 70 ' Update the PEF List
3445 71 strSQL = "SELECT DISTINCTROW [MARINE] [PEF]FROM MARINE LEFT JOIN PEF
3446 72 ON [MARINE].[PEF] = [PEF].[PEF_PK]WHERE ([PEF].[PEF_PK] Is Null)ORDER BY
3447 73 [Me.cmbPefFind.RowSource] = strSQL
3448 74
3449 75 End Sub
3450 76
3451 77 Form: frmSpecialAssignment
3452 Code
3453 1 Attribute VB_Name = "Form_frmSpecialAssignment"
3454 2 Attribute VB_Creatable = True
3455 3 Attribute VB_PredeclaredId = True
3456 4 Attribute VB_Exposed = False
3457 5 Option Compare Database
3458 6 Option Explicit
3459 7
3460 8 Private Sub cmbClassNumber_AfterUpdate()
3461 9
3462 10 Dim strConvert As String
3463 11
3464 12 Me.tbReportDate = Me.cmbClassNumber.Column(2)
3465 13 Me.tbMCC = Me.cmbClassNumber.Column(4)
3466 14
3467 15 ' Convert the fiscal year to a two digit number for RD3 File
3468 16 strConvert = CStr(Me.cmbClassNumber.Column(3))
3469 17 strConvert = Right(strConvert, 2)
3470 18 Me.tbFY = CInt(strConvert)
3471 19
3472 20 Me.tbAssignmentType = "S"
3473 21
3474 22 End Sub
3475 23
3476 24 Private Sub cmbClassNumber_Enter()
3477 25
3478 26 Dim strSQL As String
3479 27
3480 28 ' This query finds the class numbers, report dates and class
3481 29 ' numbering dates associated with the chosen school
3482 30 strSQL = "SELECT BNA_EXTRACT.ClassNumber, BNA_EXTRACT.ReportDate,
3483 31 BNA_EXTRACT.ClassConvDate, BNA_EXTRACT.MCC FROM BNA_EXTRACT INNER JOIN
3484 32 SCH_TGT_MOS ON (BNA_EXTRACT.CourseNumber_PK =
3485 33 SCH_TGT_MOS.TargetMOS_FK) AND (BNA_EXTRACT.TargetMOS_PK =
3486 34 SCH_TGT_MOS.TargetMOS_FK) WHERE BNA_EXTRACT.CourseNumber_PK =
3487 35 Form!frmSpecialAssignment!cmbCourseNumber AND SCH_TGT_MOS.AMOS_FK =
3488 36 Form!frmSpecialAssignment!tbAMOS;"
3489 37 Me.cmbClassNumber.RowSource = strSQL
3490 38
3491 39 End Sub
3492 40
3493 41 Private Sub cmbCourseNumber_AfterUpdate()
3494 42
3495 43 Me.tbAMOS = Me.cmbCourseNumber.Column(1)
3496 44
3497 45 End Sub
3498 46
3499 47 Private Sub cmbSSN_Enter()
3500 48
3501 49 Dim strSQL As String
3502 50
3503 51 strSQL = "SELECT DISTINCTROW MARINE.SSN_PK, MARINE.SSN_PK,
3504 52 MARINE.Grads FROM MARINE LEFT JOIN ASSIGNMENT ON MARINE.SSN_PK =
3505 53 ASSIGNMENT.SSN_FK WHERE ((ASSIGNMENT.SSN_FK Is Null));"
3506 54 Me.cmbSSN.RowSource = strSQL
3507 55
3508 56 End Sub
3509 57
3510 58 Private Sub cmbSSNFind_AfterUpdate()
3511 59
3512 60 Dim R As Recordset
3513 61 Set R = Me.RecordsetClone
3514 62 R.FindFirst "[SSN_FK] = " & Chr(34) & Me[cmbSSNFind] & Chr(34)
3515 63 Me.Bookmark = R.Bookmark
3516 64 Me[cmbSSNFind] = Null
3517 65 Me.Close.SetFocus
3518 66
3519 67 End Sub
3520 68
3521 69 Private Sub Delete_Click()
3522 70
3523 71 DoCmd.DoMenuItem acFormBar, acEditMenu, 8, , acMenuVer70
3524 72 DoCmd.DoMenuItem acFormBar, acEditMenu, 6, , acMenuVer70
3525 73
3526 74 Exit_Delete_Click:
3527 75 Exit Sub
3528 76
3529 77 Err_Delete_Click:
3530 78 MsgBox Err.Description
3531 79 Resume Exit_Delete_Click
3532 80
3533 81 End Sub
3534 82
3535 83 Private Sub Close_Click()
3536 84 On Error GoTo Err_Close_Click
3537 85
3538 86 DoCmd.DoMenuItem acFormBar, acEditMenu, 8, , acMenuVer70
3539 87 DoCmd.DoMenuItem acFormBar, acEditMenu, 6, , acMenuVer70
3540 88
3541 89 Exit_Close_Click:
3542 90 Exit Sub
3543 91
3544 92 Err_Close_Click:
3545 93 MsgBox Err.Description
3546 94 Resume Exit_Close_Click
3547 95
3548 96 End Sub
3549 97
3550 98 Private Sub Form_Current()
3551 99
3552 100 ' Prevents user from accidentally making special assignments to
3553 101 ' marines assigned normally
3554 102 If IsNull(Me.cmbSSN.Value) = False Then
3555 103 Me.cmbSSN.Enabled = False
3556 104 Me.cmbCourseNumber.Enabled = False
3557 105 Me.cmbClassNumber.Enabled = False
3558 106
3559 107 ' Allows the user to enter values for the fundamental equation
3560 108 Else
3561 109 Me.cmbSSN.Enabled = True
3562 110 Me.cmbCourseNumber.Enabled = True
3563 111 Me.cmbClassNumber.Enabled = True
3564 112
3565 113 End If
3566 114
3567 115 End Sub
3568 116
3569 117 Private Sub Form_Open(Cancel As Integer)
3570 118
3571 119 DoCmd.GoToRecord acDataForm, "frmSpecialAssignment", acNewRec
3572 120
3573 121 End Sub
3574 122
3575 123 Form: frmSpecialAssignmentOfUnassignedMarine
3576 Code
3577 1 Attribute VB_Name = "Form_frmSpecialAssignmentOfUnassignedMarine"
3578 2 Attribute VB_Creatable = True
3579 3 Attribute VB_PredeclaredId = True
3580 4 Attribute VB_Exposed = False
3581 5 Option Compare Database
3582 6 Option Explicit
3583 7
3584 8 Private Sub cmbClassNumber_AfterUpdate()
3585 9
3586 10 Dim strConvert As String
3587 11 Dim strSSN As String
3588 12 Dim R As Recordset
3589 13
3590 14 Me.tbReportDate = Me.cmbClassNumber.Column(2)
3591 15 Me.tbMCC = Me.cmbClassNumber.Column(4)
3592 16
3593 17 ' Convert the fiscal year to a two digit number for RD3 File
3594 18 strConvert = CStr(Me.cmbClassNumber.Column(3))
3595 19 strConvert = Right(strConvert, 2)
3596 20 Me.tbFY = CInt(strConvert)
3597 21
3598 22 Me.tbAssignmentType = "S"
3599 23
3600 24 ' Keep track of the current SSN
3601 25 strSSN = Me.cmbSSN
3602 26
3603 27 Me.tbAMOS = Me.cmbCourseNumber.Column(1)
3604 28
3605 29 ' Updates the current form and subform
3606 30 Me.Requery
3607 31 Me.Repaint
3608 32
3609 33 ' Moves the record back to the current SSN
3610 34 Set R = Me.RecordsetClone
3611 35 R.FindFirst "[SSN_FK] = " & Chr(34) & strSSN & Chr(34)
3612 36 Me.Bookmark = R.Bookmark
3613 37
3614 38 End Sub
3615 39
3616 40 Private Sub cmbClassNumber_Enter()
3617 41
3618 42 Dim strSQL As String
3619 43
3620 44 strSQL = "SELECT BNA_EXTRACT.ClassNumber, BNA_EXTRACT.ReportDate,
3621 45 BNA_EXTRACT.ClassConvDate, BNA_EXTRACT.MCC FROM BNA_EXTRACT INNER JOIN
3622 46 SCH_TGT_MOS ON (BNA_EXTRACT.CourseNumber_PK =
3623 47 SCH_TGT_MOS.TargetMOS_FK) AND (BNA_EXTRACT.TargetMOS_PK =
3624 48 SCH_TGT_MOS.TargetMOS_FK) WHERE BNA_EXTRACT.CourseNumber_PK =
3625 49 Form!frmSpecialAssignmentOfUnassignedMarine!cmbCourseNumber AND SCH_TGT_MOS.AMOS_FK =
3626 50 Form!frmSpecialAssignmentOfUnassignedMarine!tbAMOS;"
3627 51 Me.cmbClassNumber.RowSource = strSQL
3628 52
3629 53 End Sub
3630 54
3631 55 Private Sub cmbCourseNumber_AfterUpdate()
3632 56
3633 57 Me.tbAMOS = Me.cmbCourseNumber.Column(1)
3634 58
3635 59 End Sub
3636 60
3637 61 Private Sub cmbSSN_Enter()
3638 62
3639 63 Dim strSQL As String
3640 64
3641 65 strSQL = "SELECT DISTINCTROW MARINE.SSN_PK, MARINE.SSN_PK,
3642 66 MARINE.Grads FROM MARINE LEFT JOIN ASSIGNMENT ON MARINE.SSN_PK =
3643 67 ASSIGNMENT.SSN_FK WHERE ((ASSIGNMENT.SSN_FK Is Null));"
3644 68 Me.cmbSSN.RowSource = strSQL
3645 69
3646 70 End Sub

```

```

3640 44 ' This query finds the class numbers, report dates and class
3641 converting dates associated with the chosen school
3642 45 strSQL = "SELECT BNA_EXTRACT.ClassNumber_PK,
3643 BNA_EXTRACT.ClassConDate, BNA_EXTRACT.ReportDate,
3644 BNA_EXTRACT.FiscalYear_PK, BNA_EXTRACT.MCC FROM BNA_EXTRACT INNER JOIN
3645 SCH_TGT_MOS ON (BNA_EXTRACT.CourseNumber_PK =
3646 SCH_TGT_MOS.CourseNumber_PK) AND (BNA_EXTRACT.TargetMOS_PK =
3647 SCH_TGT_MOS.TargetMOS_PK) WHERE BNA_EXTRACT.CourseNumber_PK =
3648 Form!frmSpecialAssignmentOfUnassignedMarine!cmbCourseNumber AND
3649 SCH_TGT_MOS.AMOS_FK =
3650 Me.cmbClassNumber.RowSource = strSQL
3651 47
3652 48 End Sub
3653
3654 50 Private Sub cmbCourseNumber_AfterUpdate()
3655 51
3656 52 Dim strSSN As String
3657 53 Dim R As Recordset
3658 54
3659 55 strSSN = Me.cmbSSN
3660 56
3661 57 ' Keep track of the current SSN
3662 58 strSSN = Me.cmbSSN
3663 59
3664 60 Me.tblAMOS = Me!cmbCourseNumber.Column(1)
3665 61
3666 62 ' Updates the current form and subform
3667 63 Me.Requery
3668 64 Me.Repaint
3669 65
3670 66 ' Moves the record back to the current SSN
3671 67 Set R = Me.RecordsetClone
3672 68 R.FindFirst "[SSN_FK] = " & Chr(34) & strSSN & Chr(34)
3673 69 Me.Bookmark = R.Bookmark
3674 70
3675 71 End Sub
3676 72
3677 73 Private Sub cmbSSN_AfterUpdate()
3678 74
3679 75 Me.tblSQL = Me!cmbSSN.Column(1)
3680 76 Me.tblGradDate = Me!cmbSSN.Column(2)
3681 77
3682 78 End Sub
3683 79
3684 80 Private Sub cmbSSN_Enter()
3685 81
3686 82 Dim strSQL As String
3687 83
3688 84 strSQL = "SELECT DISTINCTROW MARINE.SSN_PK, MARINE.SSN,
3689 MARINE.GradDate FROM MARINE LEFT JOIN ASSIGNMENT ON MARINE.SSN_PK =
3690 ASSIGNMENT.SSN_FK WHERE (((ASSIGNMENT.SSN_FK) Is Null)):"
3691 85 Me.cmbSSN.RowSource = strSQL
3692 86
3693 87 End Sub
3694 88
3695 89 Private Sub cmbSSNFind_AfterUpdate()
3696 90
3697 91 Dim R As Recordset
3698 92 Set R = Me.RecordsetClone
3699 93 R.FindFirst "[SSN_FK] = " & Chr(34) & Me!cmbSSNFind & Chr(34)
3700 94 Me.Bookmark = R.Bookmark
3701 95 Me!cmbSSNFind = Null
3702 96 Me.Close.SetFocus
3703 97
3704 98 End Sub
3705 99
3706 100 Private Sub Delete_Click()
3707 101 On Error GoTo Err_Delete_Click
3708 102
3709 103
3710 104 DoCmd.DoMenuItem acFormBar, acEditMenu, 8, , acMenuVer70
3711 105 DoCmd.DoMenuItem acFormBar, acEditMenu, 6, , acMenuVer70
3712 106
3713 107 Exit_Delete_Click
3714 108 Exit Sub
3715 109
3716 110 Err_Delete_Click
3717 111 MsgBox Err.Description
3718 112 Resume Exit_Delete_Click
3719 113
3720 114 End Sub
3721 115
3722 116 Private Sub Close_Click()
3723 117 On Error GoTo Err_Close_Click
3724 118
3725 119 ' Close current form
3726 120 DoCmd.Close
3727 121
3728 122 Exit_Close_Click
3729 123 Exit Sub
3730 124
3731 125 Err_Close_Click
3732 126 MsgBox Err.Description
3733 127 Resume Exit_Close_Click
3734 128
3735 129 End Sub
3736 130
3737 131 Private Sub Form_Current()
3738 132
3739 133 Me!cmbSSN.Enabled = True
3740 134 Me!cmbCourseNumber.Enabled = True
3741 135 Me!cmbClassNumber.Enabled = True
3742 136
3743 137
3744 138
3745 139 End Sub
3746 140
3747 141 Private Sub Form_Open(Cancel As Integer)
3748 142
3749 143
3750 144 DoCmd.GoToRecord acDataForm,
3751 frmSpecialAssignmentOfUnassignedMarine, acNewRec
3752 145
3753 146
3754 147 End Sub
3755
3756 Form: frmUnassignedMarines
3757 Code
3758 1 Attribute VB_Name = "Form_frmUnassignedMarines"
3759 2 Attribute VB_Creatable = True
3760 3 Attribute VB_PredeclaredId = True
3761 4 Attribute VB_Exposed = False
3762 5 Option Compare Database
3763 6 Option Explicit
3764 7
3765 8 Private Sub Close_Click()
3766 9 On Error GoTo Err_Close_Click
3767 10
3768 11 Dim sDocName As String
3769 12 Dim sLinkCriteria As String

```

```

3770 13
3771 14 ' Close the current form
3772 15 DoCmd.Close
3773 16
3774 17 ' Open specified form
3775 18 sDocName = "frmAnalyzaResult"
3776 19 DoCmd.OpenForm sDocName, , , sLinkCriteria
3777 20 Exit_Close_Click
3778 21 Exit Sub
3779 22
3780 23 Err_Close_Click
3781 24 MsgBox Err.Description
3782 25 Resume Exit_Close_Click
3783 26
3784 27 End Sub
3785 28
3786 29 Private Sub cmbSSNFind_AfterUpdate()
3787 30
3788 31 Dim R As Recordset
3789 32 Set R = Me.RecordsetClone
3790 33 R.FindFirst "[SSN_PK] = " & Chr(34) & Me!cmbSSNFind & Chr(34)
3791 34 Me.Bookmark = R.Bookmark
3792 35 Me!cmbSSNFind = Null
3793 36 Me.Close.SetFocus
3794 37
3795 38 End Sub
3796 39
3797 40 Private Sub Form_Current()
3798 41
3799 42 Dim strSQL As String
3800 43 Dim rec As Recordset
3801 44 Dim db1 As Database
3802 45
3803 46 Set db1 = CurrentDb()
3804 47
3805 48 ' Calculates the number of unassigned marines.
3806 49 strSQL = "SELECT Count(PEF) AS TotalUnassigned FROM
3807 50 Set rec = db1.OpenRecordset(strSQL, dbOpenSnapshot)
3808 51 Me.tblTotalUnassigned = rec.TotalUnassigned
3809 52
3810 53 End Sub
3811 54 Private Sub btnAssign_Click()
3812 55 On Error GoTo Err_btnAssign_Click
3813 56
3814 57 Dim sDocName As String
3815 58
3816 59 sDocName = "frmSpecialAssignmentOfUnassignedMarine"
3817 60
3818 61 DoCmd.OpenForm sDocName
3819 62
3820 63 Exit_btnAssign_Click
3821 64 Exit Sub
3822 65
3823 66 Err_btnAssign_Click
3824 67 MsgBox Err.Description
3825 68 Resume Exit_btnAssign_Click
3826 69
3827 70 End Sub
3828
3829 Form: subfrmFundamentalsProperty
3830 Code
3831 1 Attribute VB_Name = "Form_subfrmFundamentalsProperty"
3832 2 Attribute VB_Creatable = True
3833 3 Attribute VB_PredeclaredId = True
3834 4 Attribute VB_Exposed = False
3835 5 Option Compare Database
3836 6 Option Explicit
3837 7 Private Sub cmbValue_FK_AfterUpdate()
3838 8
3839 9 Form!frmFundamentalsProperty!FFPropertyName_PK.SetFocus
3840 10
3841 11 End Sub
3842 12
3843 13 Private Sub cmbValue_FK_BeforeUpdate(Cancel As Integer)
3844 14
3845 15 On Error GoTo Err_cmbValue_FK_BeforeUpdate
3846 16
3847 17 Dim db As Database
3848 18 Dim rec As Recordset
3849 19 Dim strSQL As String
3850 20 Dim Convert As Variant
3851 21
3852 22 ' Places value in a form agreeable to Access (e.g. '1f' or '10')
3853 23 Convert = "" &
3854 24 Form!frmFundamentalsProperty!subfrmFundamentalsProperty!cmbValue_FK & ""
3855 25
3856 26 ' Opens the table PROPERTY_VALUE
3857 27 strSQL = "PROPERTY_VALUE"
3858 28 Set rec = CurrentDb()
3859 29 Set rec = db.OpenRecordset(strSQL, dbOpenSnapshot)
3860 30
3861 31 ' Searches for the value entered by the user in the
3862 32 rec.FindFirst "Value_FK = " & Convert
3863 33
3864 34 ' If the value is not in the PROPERTY_VALUE table, it is added.
3865 35 If rec.NoMatch = True Then
3866 36
3867 37 ' Disables the action query confirmation message
3868 38 Application.SetOption "Confirm Action Queries", False
3869 39
3870 40 ' Enters the value into the PROPERTY_VALUE table
3871 41 DoCmd.OpenQuery "qryUpdateProperty_Value"
3872 42
3873 43 ' Updates the drop down list for the combobox
3874 44 strSQL = "SELECT DISTINCT " & " FROM MARINE:"
3875 45 Form!frmFundamentalsProperty!MarineField & " FROM MARINE:"
3876 46 Form!frmFundamentalsProperty!subfrmFundamentalsProperty!cmbValue_FK.RowSou
3877 47 rce = strSQL
3878 48
3879 49 ' Enables the action query confirmation message
3880 50 Application.SetOption "Confirm Action Queries", True
3881 51 End If
3882 52
3883 53 rec.Close
3884 54
3885 55 Exit_cmbValue_FK_BeforeUpdate:
3886 56 Exit Sub
3887 57
3888 58 Err_cmbValue_FK_BeforeUpdate:
3889 59 MsgBox Err.Description
3890 60 Resume Exit_cmbValue_FK_BeforeUpdate
3891 61
3892 62
3893 63
3894 64
3895 65 End Sub
3896
3897 Form: subfrmFundamentalsPropertyList
3898 Code
3899 1 Attribute VB_Name = "Form_subfrmFundamentalsPropertyList"

```

```

3900 2 Attribute VB_Creatable = True
3901 3 Attribute VB_PredeclaredId = True
3902 4 Attribute VB_Exposed = False
3903 5 Option Compare Database
3904 6 Option Explicit
3905 7
3906 8 Private Sub cmbValue_FK_BeforeUpdate(Cancel As Integer)
3907 9 On Error GoTo Err_cmbValue_FK_BeforeUpdate
3908 10
3909 11 Dim db As Database
3910 12 Dim rec As Recordset
3911 13 Dim strSQL As String
3912 14 Dim Convert As Variant
3913 15
3914 16 'Opens the subfrmFundamentalProperty so the following code will
3915 17 DoCmd.OpenForm "subfrmFundamentalProperty", , , , acHidden
3916 18
3917 19
3918 20 'Places value in a form agreeable to Access (e.g. 'HT' or '10')
3919 21 Convert = "" & Form([subfrmFundamentalProperty])cmbValue_FK &
3920 22 'Opens the table PROPERTY_VALUE
3921 23 strSQL = "PROPERTY_VALUE"
3922 24
3923 25 Set db = CurrentDb()
3924 26 Set rec = db.OpenRecordset(strSQL, dbOpenSnapshot)
3925 27
3926 28 rec.FindFirst "Value_FK = " & Convert
3927 29 If rec.NoMatch = True Then
3928 30 'Disables the action query confirmation message
3929 31 Application.SetOption "Confirm Action Queries", False
3930 32
3931 33 DoCmd.OpenQuery "qryUpdateProperty_Value"
3932 34 strSQL = "SELECT DISTINCT * FROM MARINE;"
3933 35 Me.cmbValue_FK.RowSource = strSQL
3934 36
3935 37 'Enables the action query confirmation message
3936 38 Application.SetOption "Confirm Action Queries", True
3937 39 End If
3938 40
3939 41 rec.Close
3940 42
3941 43 'Closes the subfrmFundamentalProperty without saving changes to
3942 44 DoCmd.Close acForm, "subfrmFundamentalProperty"
3943 45
3944 46
3945 47 Exit_cmbValue_FK_BeforeUpdate:
3946 48 Exit Sub
3947 49
3948 50 Err_cmbValue_FK_BeforeUpdate:
3949 51 MsgBox Err.Description
3950 52 Application.SetOption "Confirm Action Queries", True
3951 53 Resume Exit_cmbValue_FK_BeforeUpdate
3952 54
3953 55 End Sub
3954 56
3955 57 Private Sub Value_FK_BeforeUpdate(Cancel As Integer)
3956 58 On Error GoTo Err_Value_FK_BeforeUpdate
3957 59
3958 60 Dim db As Database
3959 61 Dim rec As Recordset
3960 62 Dim strSQL As String
3961 63 Dim Convert As Variant
3962 64
3963 65 'Places value in a form agreeable to Access (e.g. 'HT' or '10')
3964 66 Convert = "" &
3965 67 Form([subfrmFundamentalProperty])ListValue_FK & ""
3966 68 'Opens the table PROPERTY_VALUE
3967 69 strSQL = "PROPERTY_VALUE"
3968 70
3969 71 Set db = CurrentDb()
3970 72 Set rec = db.OpenRecordset(strSQL, dbOpenSnapshot)
3971 73
3972 74 rec.FindFirst "Value_FK = " & Convert
3973 75 'This If Then statement checks to see if the entered value is in
3974 76 'PROPERTY_VALUE table. If not, it is added to the table.
3975 77 If rec.NoMatch = True Then
3976 78 'Disables the action query confirmation message
3977 79 Application.SetOption "Confirm Action Queries", False
3978 80
3979 81 DoCmd.OpenQuery "qryUpdateProperty_ValueList"
3980 82
3981 83 'Enables the action query confirmation message
3982 84 Application.SetOption "Confirm Action Queries", True
3983 85 End If
3984 86
3985 87 rec.Close
3986 88
3987 89 Exit_Value_FK_BeforeUpdate:
3988 90 Exit Sub
3989 91
3990 92 Err_Value_FK_BeforeUpdate:
3991 93 MsgBox Err.Description
3992 94 Application.SetOption "Confirm Action Queries", True
3993 95 Resume Exit_Value_FK_BeforeUpdate
3994 96 End Sub
3995 97
3996 98 End Sub
3997 99
3998 100
3999 101
4000 102
4001 103 Attribute VB_Name = "modFitnessDetermination"
4002 104 Option Compare Database
4003 105 Option Explicit
4004 106
4005 107 Dim lngStart, lngEnd As Long
4006 108
4007 109
4008 110 Public Function FundPropTest(FundPropName As String, SSN As String) As
4009 111
4010 112 Dim rec As Recordset, rec2 As Recordset, rec3 As Recordset
4011 113 Dim strMarField As String, strFundOperator As String, strSQL As
4012 114 Dim strMarValueSQL As String, strFundValueSQL As String, strInput As
4013 115 Dim verFundValue, verMarValue
4014 116 Dim db1 As Database
4015 117
4016 118 FundPropName = "" & FundPropName & "" 'This expression
4017 119 properly formats the argument FundPropName, by placing a ' in front of
4018 120 and behind the value found in FundPropName.
4019 121
4020 122 Set db1 = CurrentDb()
4021 123
4022 124 strSQL = "SELECT MarineField, Operator FROM FUNDAMENTAL_PROPERTY
4023 125 WHERE PropertyName_FK = " & FundPropName
4024 126 Set rec = db1.OpenRecordset(strSQL, dbOpenSnapshot)
4025 127 strMarField = rec1.MarineField
4026 128
4027 129 strFundOperator = rec1.Operator
4028 130
4029 131
4030 132 strInput = "" & SSN & ""
4031 133 strMarValueSQL = "SELECT * & strMarField & * AS [Value] FROM MARINE
4032 134 WHERE SSN_FK = " & strInput
4033 135 Set rec2 = db1.OpenRecordset(strMarValueSQL, dbOpenSnapshot)
4034 136 verMarValue = rec2.Value
4035 137
4036 138 strFundValueSQL = "SELECT Value_FK FROM FUND_PROP_VAL WHERE
4037 139 PropertyName_FK = " & FundPropName
4038 140 Set rec3 = db1.OpenRecordset(strFundValueSQL, dbOpenSnapshot)
4039 141 verFundValue = rec3.Value_FK
4040 142
4041 143 'Convert verFundValue and verMarValue to a numeric, if possible
4042 144 If IsNumeric(verFundValue) And IsNumeric(verMarValue) Then
4043 145 verFundValue = CInt(verFundValue)
4044 146 verMarValue = CInt(verMarValue)
4045 147 Else
4046 148 verFundValue = CStr(verFundValue)
4047 149 verMarValue = CStr(verMarValue)
4048 150 End If
4049 151
4050 152
4051 153 Select Case strFundOperator
4052 154 Case "="
4053 155 If verMarValue = verFundValue Then
4054 156 FundPropTest = True
4055 157 End If
4056 158 Case "<"
4057 159 If verMarValue < verFundValue Then
4058 160 FundPropTest = True
4059 161 End If
4060 162 Case "<="
4061 163 If verMarValue <= verFundValue Then
4062 164 FundPropTest = True
4063 165 End If
4064 166 Case ">"
4065 167 If verMarValue > verFundValue Then
4066 168 FundPropTest = True
4067 169 End If
4068 170 Case ">="
4069 171 If verMarValue >= verFundValue Then
4070 172 FundPropTest = True
4071 173 End If
4072 174 Case "<>"
4073 175 If verMarValue <> verFundValue Then
4074 176 FundPropTest = True
4075 177 End If
4076 178 Case "<>="
4077 179 If verMarValue <>= verFundValue Then
4078 180 FundPropTest = True
4079 181 End If
4080 182 Case "><"
4081 183 If verMarValue > verFundValue Then
4082 184 FundPropTest = True
4083 185 End If
4084 186 Case "><="
4085 187 If verMarValue >= verFundValue Then
4086 188 FundPropTest = True
4087 189 End If
4088 190 Case "<>="
4089 191 If verMarValue <>= verFundValue Then
4090 192 FundPropTest = True
4091 193 End If
4092 194 Case "<>"
4093 195 If verMarValue <> verFundValue Then
4094 196 FundPropTest = True
4095 197 End If
4096 198 Case "<="
4097 199 If verMarValue <= verFundValue Then
4098 200 FundPropTest = True
4099 201 End If
4100 202 Case ">="
4101 203 If verMarValue >= verFundValue Then
4102 204 FundPropTest = True
4103 205 End If
4104 206 Case "<>"
4105 207 If verMarValue <> verFundValue Then
4106 208 FundPropTest = True
4107 209 End If
4108 210 Case "<="
4109 211 If verMarValue <= verFundValue Then
4110 212 FundPropTest = True
4111 213 End If
4112 214 Case ">="
4113 215 If verMarValue >= verFundValue Then
4114 216 FundPropTest = True
4115 217 End If
4116 218 Case "<>"
4117 219 If verMarValue <> verFundValue Then
4118 220 FundPropTest = True
4119 221 End If
4120 222 Case "<>="
4121 223 If verMarValue <>= verFundValue Then
4122 224 FundPropTest = True
4123 225 End If
4124 226 Case "><"
4125 227 If verMarValue > verFundValue Then
4126 228 FundPropTest = True
4127 229 End If
4128 230 Case "><="
4129 231 If verMarValue >= verFundValue Then
4130 232 FundPropTest = True
4131 233 End If
4132 234 Case "<>"
4133 235 If verMarValue <> verFundValue Then
4134 236 FundPropTest = True
4135 237 End If
4136 238 Case "<="
4137 239 If verMarValue <= verFundValue Then
4138 240 FundPropTest = True
4139 241 End If
4140 242 Case ">="
4141 243 If verMarValue >= verFundValue Then
4142 244 FundPropTest = True
4143 245 End If
4144 246 Case "<>"
4145 247 If verMarValue <> verFundValue Then
4146 248 FundPropTest = True
4147 249 End If
4148 250 Case "<>="
4149 251 If verMarValue <>= verFundValue Then
4150 252 FundPropTest = True
4151 253 End If
4152 254 Case "><"
4153 255 If verMarValue > verFundValue Then
4154 256 FundPropTest = True
4155 257 End If
4156 258 Case "><="
4157 259 If verMarValue >= verFundValue Then
4158 260 FundPropTest = True
4159 261 End If
4160 262 Case "<>"
4161 263 If verMarValue <> verFundValue Then
4162 264 FundPropTest = True
4163 265 End If
4164 266 Case "<="
4165 267 If verMarValue <= verFundValue Then
4166 268 FundPropTest = True
4167 269 End If
4168 270 Case ">="
4169 271 If verMarValue >= verFundValue Then
4170 272 FundPropTest = True
4171 273 End If
4172 274 Case "<>"
4173 275 If verMarValue <> verFundValue Then
4174 276 FundPropTest = True
4175 277 End If
4176 278 Case "<>="
4177 279 If verMarValue <>= verFundValue Then
4178 280 FundPropTest = True
4179 281 End If
4180 282 Case "><"
4181 283 If verMarValue > verFundValue Then
4182 284 FundPropTest = True
4183 285 End If
4184 286 Case "><="
4185 287 If verMarValue >= verFundValue Then
4186 288 FundPropTest = True
4187 289 End If
4188 290 Case "<>"
4189 291 If verMarValue <> verFundValue Then
4190 292 FundPropTest = True
4191 293 End If
4192 294 Case "<="
4193 295 If verMarValue <= verFundValue Then
4194 296 FundPropTest = True
4195 297 End If
4196 298 Case ">="
4197 299 If verMarValue >= verFundValue Then
4198 300 FundPropTest = True
4199 301 End If
4200 302 Case "<>"
4201 303 If verMarValue <> verFundValue Then
4202 304 FundPropTest = True
4203 305 End If
4204 306 Case "<>="
4205 307 If verMarValue <>= verFundValue Then
4206 308 FundPropTest = True
4207 309 End If
4208 310 Case "><"
4209 311 If verMarValue > verFundValue Then
4210 312 FundPropTest = True
4211 313 End If
4212 314 Case "><="
4213 315 If verMarValue >= verFundValue Then
4214 316 FundPropTest = True
4215 317 End If
4216 318 Case "<>"
4217 319 If verMarValue <> verFundValue Then
4218 320 FundPropTest = True
4219 321 End If
4220 322 Case "<="
4221 323 If verMarValue <= verFundValue Then
4222 324 FundPropTest = True
4223 325 End If
4224 326 Case ">="
4225 327 If verMarValue >= verFundValue Then
4226 328 FundPropTest = True
4227 329 End If
4228 330 Case "<>"
4229 331 If verMarValue <> verFundValue Then
4230 332 FundPropTest = True
4231 333 End If
4232 334 Case "<>="
4233 335 If verMarValue <>= verFundValue Then
4234 336 FundPropTest = True
4235 337 End If
4236 338 Case "><"
4237 339 If verMarValue > verFundValue Then
4238 340 FundPropTest = True
4239 341 End If
4240 342 Case "><="
4241 343 If verMarValue >= verFundValue Then
4242 344 FundPropTest = True
4243 345 End If
4244 346 Case "<>"
4245 347 If verMarValue <> verFundValue Then
4246 348 FundPropTest = True
4247 349 End If
4248 350 Case "<="
4249 351 If verMarValue <= verFundValue Then
4250 352 FundPropTest = True
4251 353 End If
4252 354 Case ">="
4253 355 If verMarValue >= verFundValue Then
4254 356 FundPropTest = True
4255 357 End If
4256 358 Case "<>"
4257 359 If verMarValue <> verFundValue Then
4258 360 FundPropTest = True
4259 361 End If
4260 362 Case "<>="
4261 363 If verMarValue <>= verFundValue Then
4262 364 FundPropTest = True
4263 365 End If
4264 366 Case "><"
4265 367 If verMarValue > verFundValue Then
4266 368 FundPropTest = True
4267 369 End If
4268 370 Case "><="
4269 371 If verMarValue >= verFundValue Then
4270 372 FundPropTest = True
4271 373 End If
4272 374 Case "<>"
4273 375 If verMarValue <> verFundValue Then
4274 376 FundPropTest = True
4275 377 End If
4276 378 Case "<="
4277 379 If verMarValue <= verFundValue Then
4278 380 FundPropTest = True
4279 381 End If
4280 382 Case ">="
4281 383 If verMarValue >= verFundValue Then
4282 384 FundPropTest = True
4283 385 End If
4284 386 Case "<>"
4285 387 If verMarValue <> verFundValue Then
4286 388 FundPropTest = True
4287 389 End If
4288 390 Case "<>="
4289 391 If verMarValue <>= verFundValue Then
4290 392 FundPropTest = True
4291 393 End If
4292 394 Case "><"
4293 395 If verMarValue > verFundValue Then
4294 396 FundPropTest = True
4295 397 End If
4296 398 Case "><="
4297 399 If verMarValue >= verFundValue Then
4298 400 FundPropTest = True
4299 401 End If
4300 402 Case "<>"
4301 403 If verMarValue <> verFundValue Then
4302 404 FundPropTest = True
4303 405 End If
4304 406 Case "<="
4305 407 If verMarValue <= verFundValue Then
4306 408 FundPropTest = True
4307 409 End If
4308 410 Case ">="
4309 411 If verMarValue >= verFundValue Then
4310 412 FundPropTest = True
4311 413 End If
4312 414 Case "<>"
4313 415 If verMarValue <> verFundValue Then
4314 416 FundPropTest = True
4315 417 End If
4316 418 Case "<>="
4317 419 If verMarValue <>= verFundValue Then
4318 420 FundPropTest = True
4319 421 End If
4320 422 Case "><"
4321 423 If verMarValue > verFundValue Then
4322 424 FundPropTest = True
4323 425 End If
4324 426 Case "><="
4325 427 If verMarValue >= verFundValue Then
4326 428 FundPropTest = True
4327 429 End If
4328 430 Case "<>"
4329 431 If verMarValue <> verFundValue Then
4330 432 FundPropTest = True
4331 433 End If
4332 434 Case "<="
4333 435 If verMarValue <= verFundValue Then
4334 436 FundPropTest = True
4335 437 End If
4336 438 Case ">="
4337 439 If verMarValue >= verFundValue Then
4338 440 FundPropTest = True
4339 441 End If
4340 442 Case "<>"
4341 443 If verMarValue <> verFundValue Then
4342 444 FundPropTest = True
4343 445 End If
4344 446 Case "<>="
4345 447 If verMarValue <>= verFundValue Then
4346 448 FundPropTest = True
4347 449 End If
4348 450 Case "><"
4349 451 If verMarValue > verFundValue Then
4350 452 FundPropTest = True
4351 453 End If
4352 454 Case "><="
4353 455 If verMarValue >= verFundValue Then
4354 456 FundPropTest = True
4355 457 End If
4356 458 Case "<>"
4357 459 If verMarValue <> verFundValue Then
4358 460 FundPropTest = True
4359 461 End If
4360 462 Case "<="
4361 463 If verMarValue <= verFundValue Then
4362 464 FundPropTest = True
4363 465 End If
4364 466 Case ">="
4365 467 If verMarValue >= verFundValue Then
4366 468 FundPropTest = True
4367 469 End If
4368 470 Case "<>"
4369 471 If verMarValue <> verFundValue Then
4370 472 FundPropTest = True
4371 473 End If
4372 474 Case "<>="
4373 475 If verMarValue <>= verFundValue Then
4374 476 FundPropTest = True
4375 477 End If
4376 478 Case "><"
4377 479 If verMarValue > verFundValue Then
4378 480 FundPropTest = True
4379 481 End If
4380 482 Case "><="
4381 483 If verMarValue >= verFundValue Then
4382 484 FundPropTest = True
4383 485 End If
4384 486 Case "<>"
4385 487 If verMarValue <> verFundValue Then
4386 488 FundPropTest = True
4387 489 End If
4388 490 Case "<="
4389 491 If verMarValue <= verFundValue Then
4390 492 FundPropTest = True
4391 493 End If
4392 494 Case ">="
4393 495 If verMarValue >= verFundValue Then
4394 496 FundPropTest = True
4395 497 End If
4396 498 Case "<>"
4397 499 If verMarValue <> verFundValue Then
4398 500 FundPropTest = True
4399 501 End If
4400 502 Case "<>="
4401 503 If verMarValue <>= verFundValue Then
4402 504 FundPropTest = True
4403 505 End If
4404 506 Case "><"
4405 507 If verMarValue > verFundValue Then
4406 508 FundPropTest = True
4407 509 End If
4408 510 Case "><="
4409 511 If verMarValue >= verFundValue Then
4410 512 FundPropTest = True
4411 513 End If
4412 514 Case "<>"
4413 515 If verMarValue <> verFundValue Then
4414 516 FundPropTest = True
4415 517 End If
4416 518 Case "<="
4417 519 If verMarValue <= verFundValue Then
4418 520 FundPropTest = True
4419 521 End If
4420 522 Case ">="
4421 523 If verMarValue >= verFundValue Then
4422 524 FundPropTest = True
4423 525 End If
4424 526 Case "<>"
4425 527 If verMarValue <> verFundValue Then
4426 528 FundPropTest = True
4427 529 End If
4428 530 Case "<>="
4429 531 If verMarValue <>= verFundValue Then
4430 532 FundPropTest = True
4431 533 End If
4432 534 Case "><"
4433 535 If verMarValue > verFundValue Then
4434 536 FundPropTest = True
4435 537 End If
4436 538 Case "><="
4437 539 If verMarValue >= verFundValue Then
4438 540 FundPropTest = True
4439 541 End If
4440 542 Case "<>"
4441 543 If verMarValue <> verFundValue Then
4442 544 FundPropTest = True
4443 545 End If
4444 546 Case "<="
4445 547 If verMarValue <= verFundValue Then
4446 548 FundPropTest = True
4447 549 End If
4448 550 Case ">="
4449 551 If verMarValue >= verFundValue Then
4450 552 FundPropTest = True
4451 553 End If
4452 554 Case "<>"
4453 555 If verMarValue <> verFundValue Then
4454 556 FundPropTest = True
4455 557 End If
4456 558 Case "<>="
4457 559 If verMarValue <>= verFundValue Then
4458 560 FundPropTest = True
4459 561 End If
4460 562 Case "><"
4461 563 If verMarValue > verFundValue Then
4462 564 FundPropTest = True
4463 565 End If
4464 566 Case "><="
4465 567 If verMarValue >= verFundValue Then
4466 568 FundPropTest = True
4467 569 End If
4468 570 Case "<>"
4469 571 If verMarValue <> verFundValue Then
4470 572 FundPropTest = True
4471 573 End If
4472 574 Case "<="
4473 575 If verMarValue <= verFundValue Then
4474 576 FundPropTest = True
4475 577 End If
4476 578 Case ">="
4477 579 If verMarValue >= verFundValue Then
4478 580 FundPropTest = True
4479 581 End If
4480 582 Case "<>"
4481 583 If verMarValue <> verFundValue Then
4482 584 FundPropTest = True
4483 585 End If
4484 586 Case "<>="
4485 587 If verMarValue <>= verFundValue Then
4486 588 FundPropTest = True
4487 589 End If
4488 590 Case "><"
4489 591 If verMarValue > verFundValue Then
4490 592 FundPropTest = True
4491 593 End If
4492 594 Case "><="
4493 595 If verMarValue >= verFundValue Then
4494 596 FundPropTest = True
4495 597 End If
4496 598 Case "<>"
4497 599 If verMarValue <> verFundValue Then
4498 600 FundPropTest = True
4499 601 End If
4500 602 Case "<="
4501 603 If verMarValue <= verFundValue Then
4502 604 FundPropTest = True
4503 605 End If
4504 606 Case ">="
4505 607 If verMarValue >= verFundValue Then
4506 608 FundPropTest = True
4507 609 End If
4508 610 Case "<>"
4509 611 If verMarValue <> verFundValue Then
4510 612 FundPropTest = True
4511 613 End If
4512 614 Case "<>="
4513 615 If verMarValue <>= verFundValue Then
4514 616 FundPropTest = True
4515 617 End If
4516 618 Case "><"
4517 619 If verMarValue > verFundValue Then
4518 620 FundPropTest = True
4519 621 End If
4520 622 Case "><="
4521 623 If verMarValue >= verFundValue Then
4522 624 FundPropTest = True
4523 625 End If
4524 626 Case "<>"
4525 627 If verMarValue <> verFundValue Then
4526 628 FundPropTest = True
4527 629 End If
4528 630 Case "<="
4529 631 If verMarValue <= verFundValue Then
4530 632 FundPropTest = True
4531 633 End If
4532 634 Case ">="
4533 635 If verMarValue >= verFundValue Then
4534 636 FundPropTest = True
4535 637 End If
4536 638 Case "<>"
4537 639 If verMarValue <> verFundValue Then
4538 640 FundPropTest = True
4539 641 End If
4540 642 Case "<>="
4541 643 If verMarValue <>= verFundValue Then
4542 644 FundPropTest = True
4543 645 End If
4544 646 Case "><"
4545 647 If verMarValue > verFundValue Then
4546 648 FundPropTest = True
4547 649 End If
4548 650 Case "><="
4549 651 If verMarValue >= verFundValue Then
4550 652 FundPropTest = True
4551 653 End If
4552 654 Case "<>"
4553 655 If verMarValue <> verFundValue Then
4554 656 FundPropTest = True
4555 657 End If
4556 658 Case "<="
4557 659 If verMarValue <= verFundValue Then
4558 660 FundPropTest = True
4559 661 End If
4560 662 Case ">="
4561 663 If verMarValue >= verFundValue Then
4562 664 FundPropTest = True
4563 665 End If
4564 666 Case "<>"
4565 667 If verMarValue <> verFundValue Then
4566 668 FundPropTest = True
4567 669 End If
4568 670 Case "<>="
4569 671 If verMarValue <>= verFundValue Then
4570 672 FundPropTest = True
4571 673 End If
4572 674 Case "><"
4573 675 If verMarValue > verFundValue Then
4574 676 FundPropTest = True
4575 677 End If
4576 678 Case "><="
4577 679 If verMarValue >= verFundValue Then
4578 680 FundPropTest = True
4579 681 End If
4580 682 Case "<>"
4581 683 If verMarValue <> verFundValue Then
4582 684 FundPropTest = True
4583 685 End If
4584 686 Case "<="
4585 687 If verMarValue <= verFundValue Then
4586 688 FundPropTest = True
4587 689 End If
4588 690 Case ">="
4589 691 If verMarValue >= verFundValue Then
4590 692 FundPropTest = True
4591 693 End If
4592 694 Case "<>"
4593 695 If verMarValue <> verFundValue Then
4594 696 FundPropTest = True
4595 697 End If
4596 698 Case "<>="
4597 699 If verMarValue <>= verFundValue Then
4598 700 FundPropTest = True
4599 701 End If
4600 702 Case "><"
4601 703 If verMarValue > verFundValue Then
4602 704 FundPropTest = True
4603 705 End If
4604 706 Case "><="
4605 707 If verMarValue >= verFundValue Then
4606 708 FundPropTest = True
4607 709 End If
4608 710 Case "<>"
4609 711 If verMarValue <> verFundValue Then
4610 712 FundPropTest = True
4611 713 End If
4612 714 Case "<="
4613 715 If verMarValue <= verFundValue Then
4614 716 FundPropTest = True
4615 717 End If
4616 718 Case ">="
4617 719 If verMarValue >= verFundValue Then
4618 720 FundPropTest = True
4619 721 End If
4620 722 Case "<>"
4621 723 If verMarValue <> verFundValue Then
4622 724 FundPropTest = True
4623 725 End If
4624 726 Case "<>="
4625 727 If verMarValue <>= verFundValue Then
4626 728 FundPropTest = True
4627 729 End If
4628 730 Case "><"
4629 731 If verMarValue > verFundValue Then
4630 732 FundPropTest = True
4631 733 End If
4632 734 Case "><="
4633 735 If verMarValue >= verFundValue Then
4634 736 FundPropTest = True
4635 737 End If
4636 738 Case "<>"
4637 739 If verMarValue <> verFundValue Then
4638 740 FundPropTest = True
4639 741 End If
4640 742 Case "<="
4641 743 If verMarValue <= verFundValue Then
4642 744 FundPropTest = True
4643 745 End If
4644 746 Case ">="
4645 747 If verMarValue >= verFundValue Then
4646 748 FundPropTest = True
4647 749 End If
4648 750 Case "<>"
4649 751 If verMarValue <> verFundValue Then
4650 752 FundPropTest = True
4651 753 End If
4652 754 Case "<>="
4653 755 If verMarValue <>= verFundValue Then
4654 756 FundPropTest = True
4655 757 End If
4656 758 Case "><"
4657 759 If verMarValue > verFundValue Then
4658 760 FundPropTest = True
4659 761 End If
4660 762 Case "><="
4661 763 If verMarValue >= verFundValue Then
4662 764 FundPropTest = True
4663 765 End If
4664 766 Case "<>"
4665 767 If verMarValue <> verFundValue Then
4666 768 FundPropTest = True
4667 769 End If
4668 770 Case "<="
4669 771 If verMarValue <= verFundValue Then
4670 772 FundPropTest = True
4671 773 End If
4672 774 Case ">="
4673 775 If verMarValue >= verFundValue Then
4674 776 FundPropTest = True
4675 777 End If
4676 778 Case "<>"
4677 779 If verMarValue <> verFundValue Then
4678 780 FundPropTest = True
4679 781 End If
4680 782 Case "<>="
4681 783 If verMarValue <>= verFundValue Then
4682 784 FundPropTest = True
4683 785 End If
4684 786 Case "><"
4685 787 If verMarValue > verFundValue Then
4686 788 FundPropTest = True
4687 789 End If
4688 790 Case "><="
4689 791 If verMarValue >= verFundValue Then
4690 792 FundPropTest = True
4691 793 End If
4692 794 Case "<>"
4693 795 If verMarValue <> verFundValue Then
4694 796 FundPropTest = True
4695 797 End If
4696 798 Case "<="
4697 799 If verMarValue <= verFundValue Then
4698 800 FundPropTest = True
4699 801 End If
4700 802 Case ">="
4701 803 If verMarValue >= verFundValue Then
4702 804 FundPropTest = True
4703 805 End If
4704 806 Case "<>"
4705 807 If verMarValue <> verFundValue Then
4706 808 FundPropTest = True
4707 809 End If
4708 810 Case "<>="
4709 811
```

```

4160 145 'characters, therefore we subtract the position of the space
4161 146 'the length of the string. This gives us the number of
4162 characters remaining
4163 147 strLogicalEquation = LTrim(Right$(strLogicalEquation,
4164 Len(strLogicalEquation) - intSpacePos))
4165 148
4166 149 'find the next space
4167 150 intSpacePos = InStr(strLogicalEquation, " ")
4168 151 Loop
4169 152
4170 153 For j = 1 To (Ubound(strWorkingEqn))
4171 154 If strWorkingEqn(j) = "(" Then
4172 155
4173 156 Line1:
4174 157 For k = (j + 1) To (Ubound(strWorkingEqn))
4175 158 Select Case strWorkingEqn(k)
4176 159 Case ")"
4177 160 For i = (j + 1) To (k - 1)
4178 161 Select Case strWorkingEqn(i)
4179 162 Case "Or"
4180 163 boolOrOperator = True
4181 164 Case "And"
4182 165 boolAndOperator = True
4183 166 Case "Not"
4184 167 boolNotOperator = True
4185 168 Case "True"
4186 169 If strWorkingEqn(i) = "(" Then
4187 170 If boolOrOperator Then
4188 171 strWorkingEqn(i) = "False"
4189 172 boolOrOperator = False
4190 173 Else
4191 174 strWorkingEqn(i) = "True"
4192 175 End If
4193 176 Else 'Executed after the first
4194 177 bracketed property is tested and stored in the location of "i"
4195 178 If boolOrOperator Then
4196 179 strWorkingEqn(i) = "False"
4197 180 boolOrOperator = False
4198 181 End If
4199 182 If boolAndOperator Then
4200 183 strWorkingEqn(i) = "True"
4201 184 Else
4202 185 strWorkingEqn(i) = "False"
4203 186 End If
4204 187 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4205 188 boolOrOperator = False
4206 189 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4207 190 boolAndOperator = False
4208 191 End If
4209 192 Case "Not"
4210 193 If strWorkingEqn(i) = "(" Then
4211 194 If boolOrOperator Then
4212 195 strWorkingEqn(i) = "True"
4213 196 boolOrOperator = False
4214 197 Else
4215 198 strWorkingEqn(i) = "False"
4216 199 End If
4217 200 Else 'Executed after the first
4218 201 bracketed property is tested and stored in the location of "i"
4219 202 If boolOrOperator Then
4220 203 strWorkingEqn(i) = "True"
4221 204 boolOrOperator = False
4222 205 End If
4223 206 If boolAndOperator Then
4224 207 strWorkingEqn(i) = "False"
4225 208 End If
4226 209 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4227 210 boolOrOperator = False
4228 211 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4229 212 boolAndOperator = False
4230 213 End If
4231 214 Case "True" 'Do nothing
4232 215 Case Else
4233 216 If strWorkingEqn(i) = "(" Then
4234 217 If boolOrOperator Then
4235 218 strWorkingEqn(i) = "True"
4236 219 boolOrOperator = False
4237 220 Else
4238 221 strWorkingEqn(i) = "False"
4239 222 End If
4240 223 Else 'Executed after the first
4241 224 bracketed property is tested and stored in the location of "i"
4242 225 If boolOrOperator Then
4243 226 strWorkingEqn(i) = "True"
4244 227 boolOrOperator = False
4245 228 End If
4246 229 If boolAndOperator Then
4247 230 strWorkingEqn(i) = "False"
4248 231 End If
4249 232 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4250 233 boolOrOperator = False
4251 234 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4252 235 boolAndOperator = False
4253 236 End If
4254 237 Case "False" 'Do nothing
4255 238 Case Else
4256 239 If strWorkingEqn(i) = "(" Then
4257 240 If boolOrOperator Then
4258 241 strWorkingEqn(i) = "True"
4259 242 boolOrOperator = False
4260 243 Else
4261 244 strWorkingEqn(i) = "False"
4262 245 End If
4263 246 Else 'Executed after the first
4264 247 bracketed property is tested and stored in the location of "i"
4265 248 If boolOrOperator Then
4266 249 strWorkingEqn(i) = "True"
4267 250 boolOrOperator = False
4268 251 End If
4269 252 If boolAndOperator Then
4270 253 strWorkingEqn(i) = "False"
4271 254 End If
4272 255 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4273 256 boolOrOperator = False
4274 257 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4275 258 boolAndOperator = False
4276 259 End If
4277 260 Case "True" 'Do nothing
4278 261 Case Else
4279 262 If strWorkingEqn(i) = "(" Then
4280 263 If boolOrOperator Then
4281 264 strWorkingEqn(i) = "True"
4282 265 boolOrOperator = False
4283 266 Else
4284 267 strWorkingEqn(i) = "False"
4285 268 End If
4286 269 Else 'Executed after the first
4287 270 bracketed property is tested and stored in the location of "i"
4288 271 If boolOrOperator Then
4289 272 strWorkingEqn(i) = "True"
4290 273 boolOrOperator = False
4291 274 End If
4292 275 If boolAndOperator Then
4293 276 strWorkingEqn(i) = "False"
4294 277 End If
4295 278 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4296 279 boolOrOperator = False
4297 280 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4298 281 boolAndOperator = False
4299 282 End If
4300 283 Case "False" 'Do nothing
4301 284 Case Else
4302 285 If strWorkingEqn(i) = "(" Then
4303 286 If boolOrOperator Then
4304 287 strWorkingEqn(i) = "True"
4305 288 boolOrOperator = False
4306 289 Else
4307 290 strWorkingEqn(i) = "False"
4308 291 End If
4309 292 Else 'Executed after the first
4310 293 bracketed property is tested and stored in the location of "i"
4311 294 If boolOrOperator Then
4312 295 strWorkingEqn(i) = "True"
4313 296 boolOrOperator = False
4314 297 End If
4315 298 If boolAndOperator Then
4316 299 strWorkingEqn(i) = "False"
4317 300 End If
4318 301 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4319 302 boolOrOperator = False
4320 303 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4321 304 boolAndOperator = False
4322 305 End If
4323 306 Case "True" 'Do nothing
4324 307 Case Else
4325 308 If strWorkingEqn(i) = "(" Then
4326 309 If boolOrOperator Then
4327 310 strWorkingEqn(i) = "True"
4328 311 boolOrOperator = False
4329 312 Else
4330 313 strWorkingEqn(i) = "False"
4331 314 End If
4332 315 Else 'Executed after the first
4333 316 bracketed property is tested and stored in the location of "i"
4334 317 If boolOrOperator Then
4335 318 strWorkingEqn(i) = "True"
4336 319 boolOrOperator = False
4337 320 End If
4338 321 If boolAndOperator Then
4339 322 strWorkingEqn(i) = "False"
4340 323 End If
4341 324 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4342 325 boolOrOperator = False
4343 326 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4344 327 boolAndOperator = False
4345 328 End If
4346 329 Case "False" 'Do nothing
4347 330 Case Else
4348 331 If strWorkingEqn(i) = "(" Then
4349 332 If boolOrOperator Then
4350 333 strWorkingEqn(i) = "True"
4351 334 boolOrOperator = False
4352 335 Else
4353 336 strWorkingEqn(i) = "False"
4354 337 End If
4355 338 Else 'Executed after the first
4356 339 bracketed property is tested and stored in the location of "i"
4357 340 If boolOrOperator Then
4358 341 strWorkingEqn(i) = "True"
4359 342 boolOrOperator = False
4360 343 End If
4361 344 If boolAndOperator Then
4362 345 strWorkingEqn(i) = "False"
4363 346 End If
4364 347 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4365 348 boolOrOperator = False
4366 349 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4367 350 boolAndOperator = False
4368 351 End If
4369 352 Case "True" 'Do nothing
4370 353 Case Else
4371 354 If strWorkingEqn(i) = "(" Then
4372 355 If boolOrOperator Then
4373 356 strWorkingEqn(i) = "True"
4374 357 boolOrOperator = False
4375 358 Else
4376 359 strWorkingEqn(i) = "False"
4377 360 End If
4378 361 Else 'Executed after the first
4379 362 bracketed property is tested and stored in the location of "i"
4380 363 If boolOrOperator Then
4381 364 strWorkingEqn(i) = "True"
4382 365 boolOrOperator = False
4383 366 End If
4384 367 If boolAndOperator Then
4385 368 strWorkingEqn(i) = "False"
4386 369 End If
4387 370 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4388 371 boolOrOperator = False
4389 372 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4390 373 boolAndOperator = False
4391 374 End If
4392 375 Case "False" 'Do nothing
4393 376 Case Else
4394 377 If strWorkingEqn(i) = "(" Then
4395 378 If boolOrOperator Then
4396 379 strWorkingEqn(i) = "True"
4397 380 boolOrOperator = False
4398 381 Else
4399 382 strWorkingEqn(i) = "False"
4400 383 End If
4401 384 Else 'Executed after the first
4402 385 bracketed property is tested and stored in the location of "i"
4403 386 If boolOrOperator Then
4404 387 strWorkingEqn(i) = "True"
4405 388 boolOrOperator = False
4406 389 End If
4407 390 If boolAndOperator Then
4408 391 strWorkingEqn(i) = "False"
4409 392 End If
4410 393 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4411 394 boolOrOperator = False
4412 395 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4413 396 boolAndOperator = False
4414 397 End If
4415 398 Case "True" 'Do nothing
4416 399 Case Else
4417 400 If strWorkingEqn(i) = "(" Then
4418 401 If boolOrOperator Then
4419 402 strWorkingEqn(i) = "True"
4420 403 boolOrOperator = False
4421 404 Else
4422 405 strWorkingEqn(i) = "False"
4423 406 End If
4424 407 Else 'Executed after the first
4425 408 bracketed property is tested and stored in the location of "i"
4426 409 If boolOrOperator Then
4427 410 strWorkingEqn(i) = "True"
4428 411 boolOrOperator = False
4429 412 End If
4430 413 If boolAndOperator Then
4431 414 strWorkingEqn(i) = "False"
4432 415 End If
4433 416 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4434 417 boolOrOperator = False
4435 418 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4436 419 boolAndOperator = False
4437 420 End If
4438 421 Case "False" 'Do nothing
4439 422 Case Else
4440 423 If strWorkingEqn(i) = "(" Then
4441 424 If boolOrOperator Then
4442 425 strWorkingEqn(i) = "True"
4443 426 boolOrOperator = False
4444 427 Else
4445 428 strWorkingEqn(i) = "False"
4446 429 End If
4447 430 Else 'Executed after the first
4448 431 bracketed property is tested and stored in the location of "i"
4449 432 If boolOrOperator Then
4450 433 strWorkingEqn(i) = "True"
4451 434 boolOrOperator = False
4452 435 End If
4453 436 If boolAndOperator Then
4454 437 strWorkingEqn(i) = "False"
4455 438 End If
4456 439 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4457 440 boolOrOperator = False
4458 441 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4459 442 boolAndOperator = False
4460 443 End If
4461 444 Case "True" 'Do nothing
4462 445 Case Else
4463 446 If strWorkingEqn(i) = "(" Then
4464 447 If boolOrOperator Then
4465 448 strWorkingEqn(i) = "True"
4466 449 boolOrOperator = False
4467 450 Else
4468 451 strWorkingEqn(i) = "False"
4469 452 End If
4470 453 Else 'Executed after the first
4471 454 bracketed property is tested and stored in the location of "i"
4472 455 If boolOrOperator Then
4473 456 strWorkingEqn(i) = "True"
4474 457 boolOrOperator = False
4475 458 End If
4476 459 If boolAndOperator Then
4477 460 strWorkingEqn(i) = "False"
4478 461 End If
4479 462 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4480 463 boolOrOperator = False
4481 464 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4482 465 boolAndOperator = False
4483 466 End If
4484 467 Case "False" 'Do nothing
4485 468 Case Else
4486 469 If strWorkingEqn(i) = "(" Then
4487 470 If boolOrOperator Then
4488 471 strWorkingEqn(i) = "True"
4489 472 boolOrOperator = False
4490 473 Else
4491 474 strWorkingEqn(i) = "False"
4492 475 End If
4493 476 Else 'Executed after the first
4494 477 bracketed property is tested and stored in the location of "i"
4495 478 If boolOrOperator Then
4496 479 strWorkingEqn(i) = "True"
4497 480 boolOrOperator = False
4498 481 End If
4499 482 If boolAndOperator Then
4500 483 strWorkingEqn(i) = "False"
4501 484 End If
4502 485 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4503 486 boolOrOperator = False
4504 487 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4505 488 boolAndOperator = False
4506 489 End If
4507 490 Case "True" 'Do nothing
4508 491 Case Else
4509 492 If strWorkingEqn(i) = "(" Then
4510 493 If boolOrOperator Then
4511 494 strWorkingEqn(i) = "True"
4512 495 boolOrOperator = False
4513 496 Else
4514 497 strWorkingEqn(i) = "False"
4515 498 End If
4516 499 Else 'Executed after the first
4517 500 bracketed property is tested and stored in the location of "i"
4518 501 If boolOrOperator Then
4519 502 strWorkingEqn(i) = "True"
4520 503 boolOrOperator = False
4521 504 End If
4522 505 If boolAndOperator Then
4523 506 strWorkingEqn(i) = "False"
4524 507 End If
4525 508 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4526 509 boolOrOperator = False
4527 510 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4528 511 boolAndOperator = False
4529 512 End If
4530 513 Case "False" 'Do nothing
4531 514 Case Else
4532 515 If strWorkingEqn(i) = "(" Then
4533 516 If boolOrOperator Then
4534 517 strWorkingEqn(i) = "True"
4535 518 boolOrOperator = False
4536 519 Else
4537 520 strWorkingEqn(i) = "False"
4538 521 End If
4539 522 Else 'Executed after the first
4540 523 bracketed property is tested and stored in the location of "i"
4541 524 If boolOrOperator Then
4542 525 strWorkingEqn(i) = "True"
4543 526 boolOrOperator = False
4544 527 End If
4545 528 If boolAndOperator Then
4546 529 strWorkingEqn(i) = "False"
4547 530 End If
4548 531 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4549 532 boolOrOperator = False
4550 533 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4551 534 boolAndOperator = False
4552 535 End If
4553 536 Case "True" 'Do nothing
4554 537 Case Else
4555 538 If strWorkingEqn(i) = "(" Then
4556 539 If boolOrOperator Then
4557 540 strWorkingEqn(i) = "True"
4558 541 boolOrOperator = False
4559 542 Else
4560 543 strWorkingEqn(i) = "False"
4561 544 End If
4562 545 Else 'Executed after the first
4563 546 bracketed property is tested and stored in the location of "i"
4564 547 If boolOrOperator Then
4565 548 strWorkingEqn(i) = "True"
4566 549 boolOrOperator = False
4567 550 End If
4568 551 If boolAndOperator Then
4569 552 strWorkingEqn(i) = "False"
4570 553 End If
4571 554 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4572 555 boolOrOperator = False
4573 556 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4574 557 boolAndOperator = False
4575 558 End If
4576 559 Case "False" 'Do nothing
4577 560 Case Else
4578 561 If strWorkingEqn(i) = "(" Then
4579 562 If boolOrOperator Then
4580 563 strWorkingEqn(i) = "True"
4581 564 boolOrOperator = False
4582 565 Else
4583 566 strWorkingEqn(i) = "False"
4584 567 End If
4585 568 Else 'Executed after the first
4586 569 bracketed property is tested and stored in the location of "i"
4587 570 If boolOrOperator Then
4588 571 strWorkingEqn(i) = "True"
4589 572 boolOrOperator = False
4590 573 End If
4591 574 If boolAndOperator Then
4592 575 strWorkingEqn(i) = "False"
4593 576 End If
4594 577 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4595 578 boolOrOperator = False
4596 579 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4597 580 boolAndOperator = False
4598 581 End If
4599 582 Case "True" 'Do nothing
4600 583 Case Else
4601 584 If strWorkingEqn(i) = "(" Then
4602 585 If boolOrOperator Then
4603 586 strWorkingEqn(i) = "True"
4604 587 boolOrOperator = False
4605 588 Else
4606 589 strWorkingEqn(i) = "False"
4607 590 End If
4608 591 Else 'Executed after the first
4609 592 bracketed property is tested and stored in the location of "i"
4610 593 If boolOrOperator Then
4611 594 strWorkingEqn(i) = "True"
4612 595 boolOrOperator = False
4613 596 End If
4614 597 If boolAndOperator Then
4615 598 strWorkingEqn(i) = "False"
4616 599 End If
4617 600 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4618 601 boolOrOperator = False
4619 602 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4620 603 boolAndOperator = False
4621 604 End If
4622 605 Case "False" 'Do nothing
4623 606 Case Else
4624 607 If strWorkingEqn(i) = "(" Then
4625 608 If boolOrOperator Then
4626 609 strWorkingEqn(i) = "True"
4627 610 boolOrOperator = False
4628 611 Else
4629 612 strWorkingEqn(i) = "False"
4630 613 End If
4631 614 Else 'Executed after the first
4632 615 bracketed property is tested and stored in the location of "i"
4633 616 If boolOrOperator Then
4634 617 strWorkingEqn(i) = "True"
4635 618 boolOrOperator = False
4636 619 End If
4637 620 If boolAndOperator Then
4638 621 strWorkingEqn(i) = "False"
4639 622 End If
4640 623 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4641 624 boolOrOperator = False
4642 625 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4643 626 boolAndOperator = False
4644 627 End If
4645 628 Case "True" 'Do nothing
4646 629 Case Else
4647 630 If strWorkingEqn(i) = "(" Then
4648 631 If boolOrOperator Then
4649 632 strWorkingEqn(i) = "True"
4650 633 boolOrOperator = False
4651 634 Else
4652 635 strWorkingEqn(i) = "False"
4653 636 End If
4654 637 Else 'Executed after the first
4655 638 bracketed property is tested and stored in the location of "i"
4656 639 If boolOrOperator Then
4657 640 strWorkingEqn(i) = "True"
4658 641 boolOrOperator = False
4659 642 End If
4660 643 If boolAndOperator Then
4661 644 strWorkingEqn(i) = "False"
4662 645 End If
4663 646 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4664 647 boolOrOperator = False
4665 648 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4666 649 boolAndOperator = False
4667 650 End If
4668 651 Case "False" 'Do nothing
4669 652 Case Else
4670 653 If strWorkingEqn(i) = "(" Then
4671 654 If boolOrOperator Then
4672 655 strWorkingEqn(i) = "True"
4673 656 boolOrOperator = False
4674 657 Else
4675 658 strWorkingEqn(i) = "False"
4676 659 End If
4677 660 Else 'Executed after the first
4678 661 bracketed property is tested and stored in the location of "i"
4679 662 If boolOrOperator Then
4680 663 strWorkingEqn(i) = "True"
4681 664 boolOrOperator = False
4682 665 End If
4683 666 If boolAndOperator Then
4684 667 strWorkingEqn(i) = "False"
4685 668 End If
4686 669 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4687 670 boolOrOperator = False
4688 671 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4689 672 boolAndOperator = False
4690 673 End If
4691 674 Case "True" 'Do nothing
4692 675 Case Else
4693 676 If strWorkingEqn(i) = "(" Then
4694 677 If boolOrOperator Then
4695 678 strWorkingEqn(i) = "True"
4696 679 boolOrOperator = False
4697 680 Else
4698 681 strWorkingEqn(i) = "False"
4699 682 End If
4700 683 Else 'Executed after the first
4701 684 bracketed property is tested and stored in the location of "i"
4702 685 If boolOrOperator Then
4703 686 strWorkingEqn(i) = "True"
4704 687 boolOrOperator = False
4705 688 End If
4706 689 If boolAndOperator Then
4707 690 strWorkingEqn(i) = "False"
4708 691 End If
4709 692 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4710 693 boolOrOperator = False
4711 694 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4712 695 boolAndOperator = False
4713 696 End If
4714 697 Case "False" 'Do nothing
4715 698 Case Else
4716 699 If strWorkingEqn(i) = "(" Then
4717 700 If boolOrOperator Then
4718 701 strWorkingEqn(i) = "True"
4719 702 boolOrOperator = False
4720 703 Else
4721 704 strWorkingEqn(i) = "False"
4722 705 End If
4723 706 Else 'Executed after the first
4724 707 bracketed property is tested and stored in the location of "i"
4725 708 If boolOrOperator Then
4726 709 strWorkingEqn(i) = "True"
4727 710 boolOrOperator = False
4728 711 End If
4729 712 If boolAndOperator Then
4730 713 strWorkingEqn(i) = "False"
4731 714 End If
4732 715 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4733 716 boolOrOperator = False
4734 717 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4735 718 boolAndOperator = False
4736 719 End If
4737 720 Case "True" 'Do nothing
4738 721 Case Else
4739 722 If strWorkingEqn(i) = "(" Then
4740 723 If boolOrOperator Then
4741 724 strWorkingEqn(i) = "True"
4742 725 boolOrOperator = False
4743 726 Else
4744 727 strWorkingEqn(i) = "False"
4745 728 End If
4746 729 Else 'Executed after the first
4747 730 bracketed property is tested and stored in the location of "i"
4748 731 If boolOrOperator Then
4749 732 strWorkingEqn(i) = "True"
4750 733 boolOrOperator = False
4751 734 End If
4752 735 If boolAndOperator Then
4753 736 strWorkingEqn(i) = "False"
4754 737 End If
4755 738 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4756 739 boolOrOperator = False
4757 740 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4758 741 boolAndOperator = False
4759 742 End If
4760 743 Case "False" 'Do nothing
4761 744 Case Else
4762 745 If strWorkingEqn(i) = "(" Then
4763 746 If boolOrOperator Then
4764 747 strWorkingEqn(i) = "True"
4765 748 boolOrOperator = False
4766 749 Else
4767 750 strWorkingEqn(i) = "False"
4768 751 End If
4769 752 Else 'Executed after the first
4770 753 bracketed property is tested and stored in the location of "i"
4771 754 If boolOrOperator Then
4772 755 strWorkingEqn(i) = "True"
4773 756 boolOrOperator = False
4774 757 End If
4775 758 If boolAndOperator Then
4776 759 strWorkingEqn(i) = "False"
4777 760 End If
4778 761 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4779 762 boolOrOperator = False
4780 763 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4781 764 boolAndOperator = False
4782 765 End If
4783 766 Case "True" 'Do nothing
4784 767 Case Else
4785 768 If strWorkingEqn(i) = "(" Then
4786 769 If boolOrOperator Then
4787 770 strWorkingEqn(i) = "True"
4788 771 boolOrOperator = False
4789 772 Else
4790 773 strWorkingEqn(i) = "False"
4791 774 End If
4792 775 Else 'Executed after the first
4793 776 bracketed property is tested and stored in the location of "i"
4794 777 If boolOrOperator Then
4795 778 strWorkingEqn(i) = "True"
4796 779 boolOrOperator = False
4797 780 End If
4798 781 If boolAndOperator Then
4799 782 strWorkingEqn(i) = "False"
4800 783 End If
4799 784 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4801 785 boolOrOperator = False
4802 786 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4803 787 boolAndOperator = False
4804 788 End If
4805 789 Case "False" 'Do nothing
4806 790 Case Else
4807 791 If strWorkingEqn(i) = "(" Then
4808 792 If boolOrOperator Then
4809 793 strWorkingEqn(i) = "True"
4810 794 boolOrOperator = False
4811 795 Else
4812 796 strWorkingEqn(i) = "False"
4813 797 End If
4814 798 Else 'Executed after the first
4815 799 bracketed property is tested and stored in the location of "i"
4816 800 If boolOrOperator Then
4817 801 strWorkingEqn(i) = "True"
4818 802 boolOrOperator = False
4819 803 End If
4820 804 If boolAndOperator Then
4821 805 strWorkingEqn(i) = "False"
4822 806 End If
4823 807 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4824 808 boolOrOperator = False
4825 809 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4826 810 boolAndOperator = False
4827 811 End If
4828 812 Case "True" 'Do nothing
4829 813 Case Else
4830 814 If strWorkingEqn(i) = "(" Then
4831 815 If boolOrOperator Then
4832 816 strWorkingEqn(i) = "True"
4833 817 boolOrOperator = False
4834 818 Else
4835 819 strWorkingEqn(i) = "False"
4836 820 End If
4837 821 Else 'Executed after the first
4838 822 bracketed property is tested and stored in the location of "i"
4839 823 If boolOrOperator Then
4840 824 strWorkingEqn(i) = "True"
4841 825 boolOrOperator = False
4842 826 End If
4843 827 If boolAndOperator Then
4844 828 strWorkingEqn(i) = "False"
4845 829 End If
4846 830 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4847 831 boolOrOperator = False
4848 832 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4849 833 boolAndOperator = False
4850 834 End If
4851 835 Case "False" 'Do nothing
4852 836 Case Else
4853 837 If strWorkingEqn(i) = "(" Then
4854 838 If boolOrOperator Then
4855 839 strWorkingEqn(i) = "True"
4856 840 boolOrOperator = False
4857 841 Else
4858 842 strWorkingEqn(i) = "False"
4859 843 End If
4860 844 Else 'Executed after the first
4861 845 bracketed property is tested and stored in the location of "i"
4862 846 If boolOrOperator Then
4863 847 strWorkingEqn(i) = "True"
4864 848 boolOrOperator = False
4865 849 End If
4866 850 If boolAndOperator Then
4867 851 strWorkingEqn(i) = "False"
4868 852 End If
4869 853 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4870 854 boolOrOperator = False
4871 855 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4872 856 boolAndOperator = False
4873 857 End If
4874 858 Case "True" 'Do nothing
4875 859 Case Else
4876 860 If strWorkingEqn(i) = "(" Then
4877 861 If boolOrOperator Then
4878 862 strWorkingEqn(i) = "True"
4879 863 boolOrOperator = False
4880 864 Else
4881 865 strWorkingEqn(i) = "False"
4882 866 End If
4883 867 Else 'Executed after the first
4884 868 bracketed property is tested and stored in the location of "i"
4885 869 If boolOrOperator Then
4886 870 strWorkingEqn(i) = "True"
4887 871 boolOrOperator = False
4888 872 End If
4889 873 If boolAndOperator Then
4890 874 strWorkingEqn(i) = "False"
4891 875 End If
4892 876 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4893 877 boolOrOperator = False
4894 878 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4895 879 boolAndOperator = False
4896 880 End If
4897 881 Case "False" 'Do nothing
4898 882 Case Else
4899 883 If strWorkingEqn(i) = "(" Then
4900 884 If boolOrOperator Then
4901 885 strWorkingEqn(i) = "True"
4902 886 boolOrOperator = False
4903 887 Else
4904 888 strWorkingEqn(i) = "False"
4905 889 End If
4906 890 Else 'Executed after the first
4907 891 bracketed property is tested and stored in the location of "i"
4908 892 If boolOrOperator Then
4909 893 strWorkingEqn(i) = "True"
4910 894 boolOrOperator = False
4911 895 End If
4912 896 If boolAndOperator Then
4913 897 strWorkingEqn(i) = "False"
4914 898 End If
4915 899 LogOrTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4916 900 boolOrOperator = False
4917 901 LogAndTest(CBool(strWorkingEqn(i)), CBool(strWorkingEqn(j)))
4918 902 boolAndOperator = False
4919 903 End If
4920 904 Case "True" 'Do nothing
4921 905 Case Else
4922 906 If strWorkingEqn(i) = "(" Then
4923 907 If boolOrOperator Then
4924 908 strWorkingEqn(i) = "True"
4925 909 boolOrOperator = False
4926 910 Else
4927 911 strWorkingEqn(i) = "False"
4928 912 End If
4929 913 Else 'Executed after the first
493
```

84

```

4680 596
4681 597 recMarClsRtClose
4682 598
4683 599 ' Prepare the AMPL_rdm2.dtl data file
4684 600 Application.SetOption "Confirm Action Queries", False
4685 601 db1.Execute "DELETE * FROM TEMP_FIT;"
4686 602 ' Specify the number of marines
4687 603 db1.Execute "INSERT INTO TEMP_FIT (SSN_PK, MarineIndex) VALUES
4688 (param totalMarines, : ) & recMarine.RecordCount & 1);"
4689 604 db1.Execute "INSERT INTO TEMP_FIT (SSN_PK) VALUES (:);"
4690 605 ' Specify the number of school classes
4691 606 db1.Execute "INSERT INTO TEMP_FIT (SSN_PK, MarineIndex) VALUES
4692 (param totalClasses, : ) & recClass.RecordCount & 1);"
4693 607 db1.Execute "INSERT INTO TEMP_FIT (SSN_PK) VALUES (:);"
4694 608 ' Specify the class demand
4695 609 db1.Execute "INSERT INTO TEMP_FIT (SSN_PK) VALUES (param demand
4696 610 db1.Execute "INSERT INTO TEMP_FIT (ClassIndex, FK_Fitness) SELECT
4697 CLASS.ClassIndex, CLASS.Quota FROM CLASS;"
4698 611 db1.Execute "INSERT INTO TEMP_FIT (SSN_PK) VALUES (:);"
4699 612 ' Specify the class penalty
4700 613 db1.Execute "INSERT INTO TEMP_FIT (SSN_PK) VALUES (param penalty
4701 614 db1.Execute "INSERT INTO TEMP_FIT (ClassIndex, FK_Fitness) SELECT
4702 PENALTY.ClassIndex, FK_PENALTY.Penalty FROM PENALTY;"
4703 615 db1.Execute "INSERT INTO TEMP_FIT (SSN_PK) VALUES (:);"
4704 616 DoCmd.TransferText acExportFixed, "Rdm2_Data Export Specification",
4705 TEMP_FIT, "C:\RDMM\AmplData\rdm2.dtl"
4706 617
4707 618 ' Prepare the AMPL_rdm2.dtl data file
4708 619 db1.Execute "DELETE * FROM TEMP_FIT;"
4709 620 db1.Execute "INSERT INTO TEMP_FIT SELECT * FROM MAR_CLS_FIT;"
4710 621 db1.Execute "ALTER TABLE TEMP_FIT DROP COLUMN SSN_PK;"
4711 622 db1.Execute "ALTER TABLE TEMP_FIT ADD COLUMN SSN_PK text;"
4712 623 db1.Execute "INSERT INTO TEMP_FIT (SSN_PK) VALUES (param fitness
4713 default 0);"
4714 624 db1.Execute "SELECT * INTO AMPL_TEMP FROM TEMP_FIT ORDER BY SSN_PK"
4715 625 db1.Execute "INSERT INTO AMPL_TEMP (SSN_PK) VALUES (:);"
4716 626 DoCmd.TransferText acExportFixed, "Rdm2_Data Export Specification",
4717 AMPL_TEMP, "C:\RDMM\AmplData\rdm2.dtl"
4718 627 Application.SetOption "Confirm Action Queries", True
4719 628
4720 629 recMarine.Close
4721 630 recClass.Close
4722 631 ' Remove the created tables
4723 632 db1.Execute "DROP TABLE TEMP_FIT;"
4724 633 db1.Execute "DROP TABLE AMPL_TEMP;"
4725 634 db1.Execute "Drop Table TEMP_FIT;"
4726 635 db1.Close
4727 636 ' Remove the meter from the status bar
4728 637 varRetVal = SysCmd(scSysCmdRemoveMeter)
4729 638
4730 639 End Sub
4731 640
4732 641
4733 642 Public Sub Quota_Penalty(varUpdateBound As Variant)
4734 643 Dim varDate
4735 644 Dim rec As Recordset, rec1 As Recordset
4736 645 Dim strSQL As String, strConvert As String
4737 646 Dim i As Integer
4738 647 Dim db1 As Database
4739 648
4740 649
4741 650 ' Create new class quota file
4742 651 Application.SetOption "Confirm Action Queries", False
4743 652
4744 653 ' Delete the old penalty and class files
4745 654 DoCmd.RunSQL "DELETE * FROM PENALTY"
4746 655 DoCmd.RunSQL "DELETE * FROM CLASS"
4747 656
4748 657 ' Query generates new class quota
4749 658 DoCmd.SetWarnings (False)
4750 659 DoCmd.OpenQuery "qryNewQuota"
4751 660 DoCmd.SetWarnings (True)
4752 661
4753 662 ' Delete any values < 0
4754 663 DoCmd.RunSQL "DELETE * FROM CLASS WHERE Quota <= 0;"
4755 664
4756 665 Application.SetOption "Confirm Action Queries", True
4757 666
4758 667 ' Create index for the class table and format the FY for RD3 file
4759 668 Set db1 = CurrentDB()
4760 669 strSQL = "SELECT CLASS, SCHOOL.PenaltyFactor FROM SCHOOL INNER JOIN
4761 CLASS ON (SCHOOL.AMOS_PK = CLASS.AMOS_PK) AND (SCHOOL.CourseNumber_PK =
4762 CLASS.CourseNumber_PK)"
4763 670 Set rec = db1.OpenRecordset(strSQL, dbOpenDynaset)
4764 671 rec.MoveLast
4765 672
4766 673 If rec.RecordCount > 0 Then
4767 674 rec.MoveFirst
4768 675 For i = 1 To rec.RecordCount
4769 676 rec.Edit
4770 677 rec.ClassIndex = i
4771 678 ' Converts the fiscal year to a two digit number for RD3 file
4772 679 strConvert = CStr(rec.FiscalYear_PK)
4773 680 strConvert = Right(strConvert, 2)
4774 681 rec.FiscalYear_PK = CInt(strConvert)
4775 682 rec.Update
4776 683 rec.MoveNext
4777 684 Next i
4778 685 End If
4779 686
4780 687 ' Create penalty file
4781 688 ' First, update the index from the CLASS table
4782 689 Application.SetOption "Confirm Action Queries", False
4783 690 DoCmd.OpenQuery "qryPenaltyIndex"
4784 691 Application.SetOption "Confirm Action Queries", True
4785 692 ' Create a record set of the PENALTY table
4786 693 strSQL = "SELECT * FROM PENALTY"
4787 694 Set rec1 = db1.OpenRecordset(strSQL, dbOpenDynaset)
4788 695 If IsNull(varUpdateBound) > False Then
4789 696 ' Format this date for use in the DateDiff function
4790 697 varUpdateBound = Format(varUpdateBound, "####/##/##")
4791 698
4792 699 Module: modFitnessDetermination Page: 16
4793 700
4794 701 If rec.RecordCount > 0 Then
4795 702 rec.MoveFirst
4796 703 For i = 1 To rec.RecordCount
4797 704 rec1.Edit
4798 705 ' Get report date from class record
4799 706 varDate = rec1.ReportDate
4800 707 ' Format the report date for use in the DateDiff function
4801 708 varDate = Format(varDate, "####/##/##")
4802 709 ' Make the penalty the difference between the two values
4803 710 ' This makes the penalty linear, based on date
4804 711 rec1.Penalty = DateDiff("d", varDate, varUpdateBound) *
4805 712 rec1.PenaltyFactor / 24
4806 713 rec1.Update
4807 714 rec1.MoveNext
4808 715 Next i
4809 716 End If

```



```

4939 78 DoCmd.RunSQL "DELETE DISTINCTROW MARINE.* FROM MARINE
4940 INNER JOIN ASSIGNMENT ON MARINE.SSN_PK = ASSIGNMENT.SSN_FK WHERE
4941 ASSIGNMENT.ReportDate < " & varTodaysDate & "
4942 79
4943 80 Select Case MsgBox("Delete unassigned Marines from the database?",
4944 vbYesNo + vbQuestion, "Delete Unassigned Marines")
4945 81 Case 6 " If yes is clicked
4946 82 " Deletes entries in the Marine table without a corresponding
4947 entry in the Assignment table
4948 83 DoCmd.RunSQL "DELETE DISTINCTROW MARINE.* FROM MARINE

```

```

4949 FROM MARINE LEFT JOIN ASSIGNMENT ON MARINE.SSN_PK = ASSIGNMENT.SSN_FK
4950 WHERE (((ASSIGNMENT.SSN_FK) Is Null));"
4951 84 Case 7 " If no is clicked
4952 85 " No action taken
4953 86 End Select
4954 87
4955 88 End Sub

```

## LIST OF REFERENCES

- [1] USMC, Manpower Management Information Systems Branch, Manpower and Reserve Affairs Department. "Statement of Work for the USMC Manpower Model Modernization: Reengineering of the Enlisted Assignment Model and Recruit Distribution Model," September, 1997.
- [2] Decision Support Associates Inc. *Users Manual-Recruit Distribution Model*. Developed In-house, Decision Support Associates Inc., 1993.
- [3] C.J. Date. *An Introduction to Database Systems*. Reading, MA: Addison-Wesley, 1992.
- [4] J. Dumas. *Designing User Interfaces for Software*. Englewood Cliffs, NJ: Prentice-Hall, 1988.
- [5] CPLEX, A Division of ILOG. CPLEX Home Page, 1998. [<http://www.cplex.com>].
- [6] R.H. Sprague. "A Framework for the Development of Decision Support Systems." *MIS Quarterly* 4, Pages 1-26, 1980.
- [7] Logic Works, Inc. "Logic Works BPWin Tutorial Guide," 1994-1995.
- [8] E.F. Codd. "A Relational Model of Data for Large Shared Databanks." *Communications of the ACM* 13, June 1970.
- [9] R. Smith, D. Sussman. *Beginning Access 97 VBA Programming*. Wrox Press, 1997.
- [10] R. Fourer, D. Gay, B. Kernighan. *AMPL: A Modeling Language for Mathematical Programming*. Boyd & Fraser Publishing Company, 1993.
- [11] B. Render, R. Stair. *Quantitative Analysis for Management*. Prentice-Hall Inc., 1997.



## INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center ..... 2  
     8725 John J. Kingman Rd., Ste 0944  
     Fort Belvoir, VA 22060-6218
  
2. Dudley Knox Library ..... 2  
     Naval Postgraduate School  
     411 Dyer Rd.  
     Monterey, CA 93943-5101
  
3. Professor Hemant Bhargava (Code SM/BH) ..... 7  
     Systems Management Department  
     Naval Postgraduate School  
     Monterey, CA 93940-5000
  
4. Professor Suresh Sridhar (Code SM/SR) ..... 1  
     Systems Management Department  
     Naval Postgraduate School  
     Monterey, CA 93943-5000
  
5. Professor Daniel Dolk (Code SM/Dk) ..... 1  
     Systems Management Department  
     Naval Postgraduate School  
     Monterey, CA 93943-5000
  
6. LT Kevin J. Snoap ..... 2  
     541 Westway Drive N.W.  
     Grand Rapids, MI 49544